

# **A MICROCOMPUTER IMPLEMENTATION OF THE NOISE GENERATORS FOR THE TROPO-CHANNEL SIMULATOR**

A Thesis Submitted  
In Partial Fulfilment of the Requirements  
for the Degree of

**MASTER OF TECHNOLOGY**

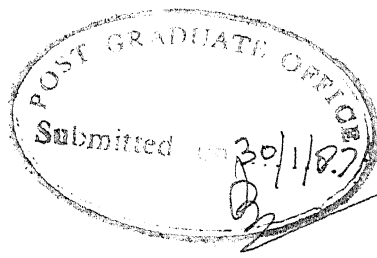
*by*

Major D. S. RANDHAWA

*to the*

DEPARTMENT OF ELECTRICAL ENGINEERING  
**INDIAN INSTITUTE OF TECHNOLOGY, KANPUR**

JANUARY, 1987



CERTIFICATE

This is to certify that the present work entitled  
'A Microcomputer Implementation of Random Noise Generators  
for the Tropo-channel Simulator' by Major D.S. Randhawa  
has been done under my supervision and has not been  
submitted elsewhere for a degree.

January, 1987

*K R. Srivathsan*  
( K.R. Srivathsan )  
Assistant Professor  
Electrical Engineering Department  
Indian Institute of Technology  
KANPUR

3 NOV 1987  
CENTRAL LIBRARY  
-----  
No. A 98553

EE-1987-M-RAN-MIC

Thesis  
621.38411  
R158m

## ACKNOWLEDGEMENTS

I express my deep appreciation and gratitude to my thesis supervisor, Dr. K.R. Srivathsan, for preparing me to take up the challenge of the present work. But for his sustained guidance at each step, the present work would never have been possible.

I would like to express my sincere thanks to the Research Engineers of ACES in general and Mr. Prem Malhotra and Mr. K.S. Ananthakrishnan in particular for their valuable guidance and help.

My thanks are also due to my friends, in particular Mr. Y.S. Ramakrishna, Major S.L. Kapoor and Mr. Sajan Jacob who were good company to work with in the same lab.

Finally, I would like to thank Mr. J.S. Rawat for some excellent typing work.

Major D.S. Randhawa

## ABSTRACT

A Tropo-channel has a randomly time varying impulse response. This time varying impulse response is equivalently characterized by the Scattering function or the Delay power spectrum of the multipath fading encountered in the channel. A Troposcatter channel simulator is an instrument which can produce the randomly time varying impulse response under controlled conditions in the laboratory. Such a simulator is a relatively inexpensive tool for testing channel-modems.

A significant part of the Tropo-simulator is the bank of random Gaussian noise generators, each having controlled spectral characteristics. These noise generators are used to produce the random fading of delayed replica of the input signal. In this work, a brief review of the block diagram of a Tropo-channel simulator is presented. This is followed by a review of the methods involved in generating random Gaussian samples, with emphasis on those useful for real time hardware implementation. Subsequently a detailed hardware implementation of the controlled Gaussian noise generators using a dual processor architecture is described. Based on it's performance capabilities suggestions are made to provide faster and more accurate implementation.

## TABLE OF CONTENTS

	Page
CHAPTER 1 THE TROPOSCATTER CHANNEL SIMULATOR	1
1.1 Introduction	1
1.2 Types of Channel Simulators	1
1.3 The Troposcatter Channel Model	2
1.4 Brief Description of the Simulator	7
1.5 Thesis Outline	9
CHAPTER 2 THE ALGORITHM OF THE NOISE GENERATOR	10
2.1 Generation of Normal Variates	10
2.1.1 Central Limit Theorem Approach	10
2.1.2 Normal Variates from Uniform Deviates	11
2.1.3 On Microcomputer Implementation	13
2.2 Generation of Uniform Deviates	14
2.2.1 Considerations	14
2.2.2 Table Look up Method	15
2.2.3 Pseudo Random Number Generator	16
2.3 Calculation of Random Numbers	22
2.4 Spectral Shaping of the Random Noise	24

CHAPTER 3	HARDWARE IMPLEMENTATION	31
3.1	The Noise Generator	31
3.2	The White Gaussian Noise Generator Card	32
3.2.1	The Single Chip Microcomputer (8741A)	32
3.2.2	Interfacing the Multiplier Accumulator, TDC 1008	36
3.2.3	Decoder Logic for White Gaussian Noise Generator Card	39
3.3	The Output Card	41
CHAPTER 4	SOFTWARE IMPLEMENTATION	45
4.1	Software for the White Gaussian Noise Generator	45
4.1.1	Implementation of the PRBSGs	45
4.1.2	Accessing the Look up Table	50
4.2	Software for the First Order Digital Filters	51
CHAPTER 5	CONCLUSION	54
REFERENCES		

## LIST OF FIGURES

Figure	Caption	Page
1.3.1	A Typical Troposcatter Communications Link	2
1.3.2	A Typical Scattering Function	5
1.3.3	A Tapped Delay Line Model	6
1.4.1	Block Diagram of the Tropo-channel simulator	8
2.2.1	Example of Feedback shift registers	17
2.2.2	An Implementation of a 34 bit PRBSG under Interrupt Control.	19
2.3.1	The Normal p.d.f. curve	22
2.3.2	Normal Distribution Function Plot	23
2.3.3	A Plot of Random Numbers of Table 2	23a
2.4.1	Examples of Power Spectra	24
2.4.2	First Order Digital Filter	28
3.1.1	Block Diagram of the Noise Generator	31
3.2.1	Block Diagram of the white Gaussian Noise Generator Card	33
3.2.2	The Buffer Structure of 8741A	34
3.2.3	Logic Diagram of TDC 1008	37
3.2.4	Data Word for 8212	39
3.2.5	Decoder Logic for the white Gaussian Noise Generator Card	40
3.3.1	Layout of the output card	42
3.3.2	Decoder Logic for the output card	43
4.1.1	Algorithm for white Gaussian Noise Generation	46
4.1.2	Organisation of Internal RAM (8741A)	47
4.1.3	Feedback logic of PRBSG	47
4.1.4	Eight PRBSGs in RAM Area of 8741A	49
4.2.1	Algorithm for the Digital Filtering	53



## CHAPTER 1

### THE TROPOSCATTER CHANNEL SIMULATOR

#### 1.1 INTRODUCTION:

A Troposcatter Channel Simulator is an instrument which simulates tropospheric channel characteristics under laboratory conditions. It's effectiveness thus directly depends on the accuracy with which the troposcatter model has been implemented

A simulator is primarily used to test modems (modulator . demodulator) under repeatable channel behaviour with controllable channel parameters. This permits the evaluation of modem's capabilities. In the absence of a simulator the testing must be done over actual tropospheric links. However, the physical separation of the transmitter and receiver and the non-stationary behaviour of the channel make the scope of testing very limited. Also, the whole procedure is costly.

#### 1.2 TYPES OF CHANNEL SIMULATORS:

'Stored Channel' and the 'Statistical Simulator' are the two type of simulators [ 9]. In the former method prerecorded gain fluctuations are replayed to reproduce the channel behaviour. On the other hand the statistical simulator is based on the modelling of the statistical behaviour of the channel and implemented in hardware in the laboratory. This

has the great advantage of simulating any type of channel, for example line of sight (LOS) flat fading or fading dispersive characteristics encountered in tropospheric communications.

### 1.3 THE TROPOSCATTER CHANNEL MODEL:

A typical troposcatter communications link is shown in Figure 1.3.1.

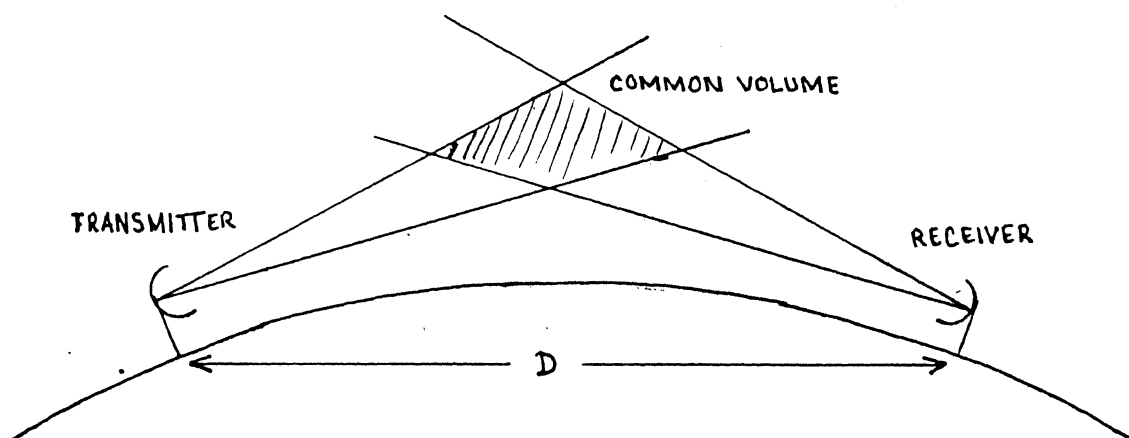


Fig. 1.3.1: A Typical Troposcatter Communications Link.

Link distances upto a few hundred Kms are typical. Frequencies in the lower microwave region are normally used. High power transmitters (1 KW to 20 KW) are used along with sensitive receivers and high gain parabolic dish antennas to overcome weak signals corrupted by additive noise and multiplicative perturbations.

As per Scattering theory the transmitted energy is scattered due to blobs in the common volume between the

transmit and receive beams. A portion of this energy is received by the receiver. Due to large perturbations in the blobs the received signal exhibits random variations caused by fast fading, also known as short term variations ranging from few nsec to minutes. The received signal is also affected by long term fading ranging from few hours to years. This is also referred to as slow fading. Nothing much can be done about it except to provide greater fade margins.

To develop the Troposcatter channel model its low pass description is considered. The signal at the receiving antenna is given by [9.]

$$\tilde{r}(t) = \int_{-\infty}^{\infty} \tilde{s}(t-\tau) \tilde{h}(\tau, t) d\tau + \tilde{n}(t) \quad (1.3.1)$$

where,  $\tilde{s}(t)$  is the complex envelope of the transmitted signal,  $\tilde{r}(t)$  is the complex envelope of the received signal,  $\tilde{h}(\tau, t)$  is the random time varying low pass equivalent impulse response of the Tropo-channel, and  $\tilde{n}(t)$  is the complex Gaussian white noise.

The channel impulse response  $\tilde{h}(\tau, t)$  is usually assumed to be the result of a wide sense stationary uncorrelated scattering [WSSUS] process in the channel. This implies that  $\tilde{h}(\tau, t)$  is a stationary process in  $t$  and that for different

values of  $\tau$ , say  $\tau_1$  and  $\tau_2$  ( $\tau_1 \neq \tau_2$ ),  $\tilde{h}(\tau_1, t)$  and  $\tilde{h}(\tau_2, t)$  are uncorrelated, i.e.,

$$E[\tilde{h}(\tau_1, t) \tilde{h}(\tau_2, t)] = 0 \quad (1.3.2)$$

Such a model allows us to describe the channel impulse response behaviour  $\tilde{h}(\tau, t)$  in terms of its power spectrum. Defining the autocorrelation of  $\tilde{h}(\tau, t)$  as

$$Q(\Delta t, \tau) = E[\tilde{h}^*(\tau, t) \tilde{h}(\tau, t + \Delta t)] \quad (1.3.3)$$

which is also equal to the time average  $\langle \tilde{h}^*(\tau, t) \tilde{h}(\tau, t + \Delta t) \rangle$  due to ergodicity of the WSSUS process. Then the power spectrum of the  $\tilde{h}(\tau, t)$  is given by

$$S(f, \tau) = \int_{-\infty}^{\infty} Q(\Delta t, \tau) e^{-j2\pi f \Delta t} d(\Delta t) \quad (1.3.4)$$

The function  $S(f, \tau)$  is more popularly known as the Scattering function of the channel. A typical Scattering function plot, shown in Figure 1.3.2, describes intuitively how much a carrier pulse of very short duration as compared to ' $\tau$ ' spread of  $S(f, \tau)$  is spread out both in time and frequency at the receiver end. As such the tropo-channel is also known as a delay-doppler channel.

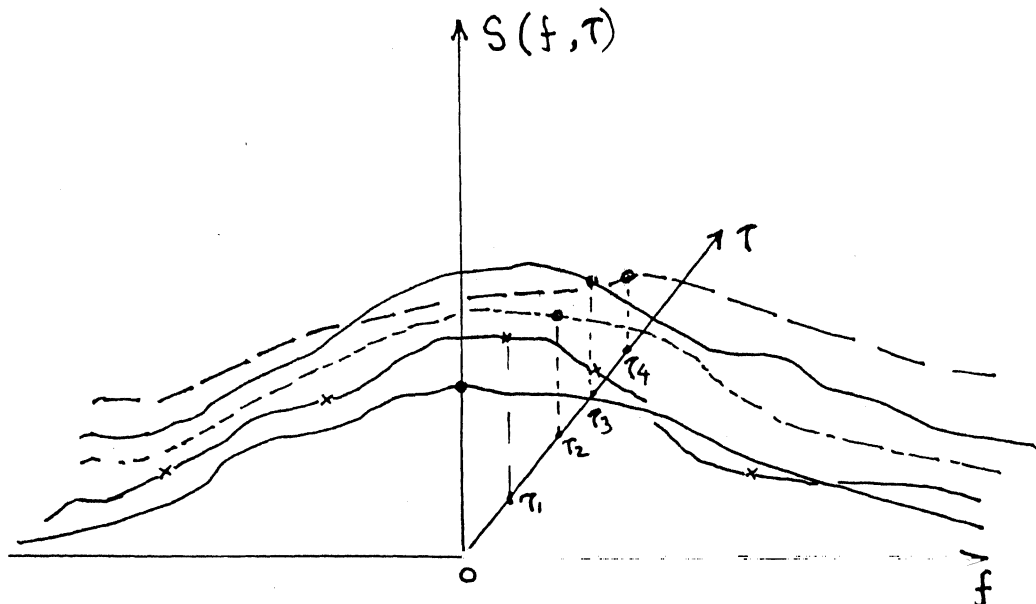


Fig. 1.3.2: A Typical Scattering Function.

Let

$$\tilde{h}_k(t) = \tilde{h}(\tau, t), \text{ for } \tau = kT \quad (1.3.5)$$

Since  $\tilde{h}(\tau, t)$  is causal, we can assume  $\tilde{h}_k(t)$  to be zero for  $k \leq 0$ . For  $T$  much smaller than the spread of  $\tilde{h}(\tau, t)$  in  $\tau$ , and finite time spread of the impulse response,  $\tilde{h}(\tau, t)$  is equivalently described by the sample functions  $\tilde{h}_0(t), \tilde{h}_1(t), \dots, \tilde{h}_N(t)$ . Then, Eqn. (1.3.1) can be rewritten as a summation.

$$\tilde{r}(t) = \sum_{k=0}^N \tilde{s}(t-kT) \tilde{h}_k(t) + \tilde{n}(t) \quad (1.3.6)$$

This discrete description provides the justification for the tapped delay line model [9] of the tropo-channel shown in Figure 1.3.3.

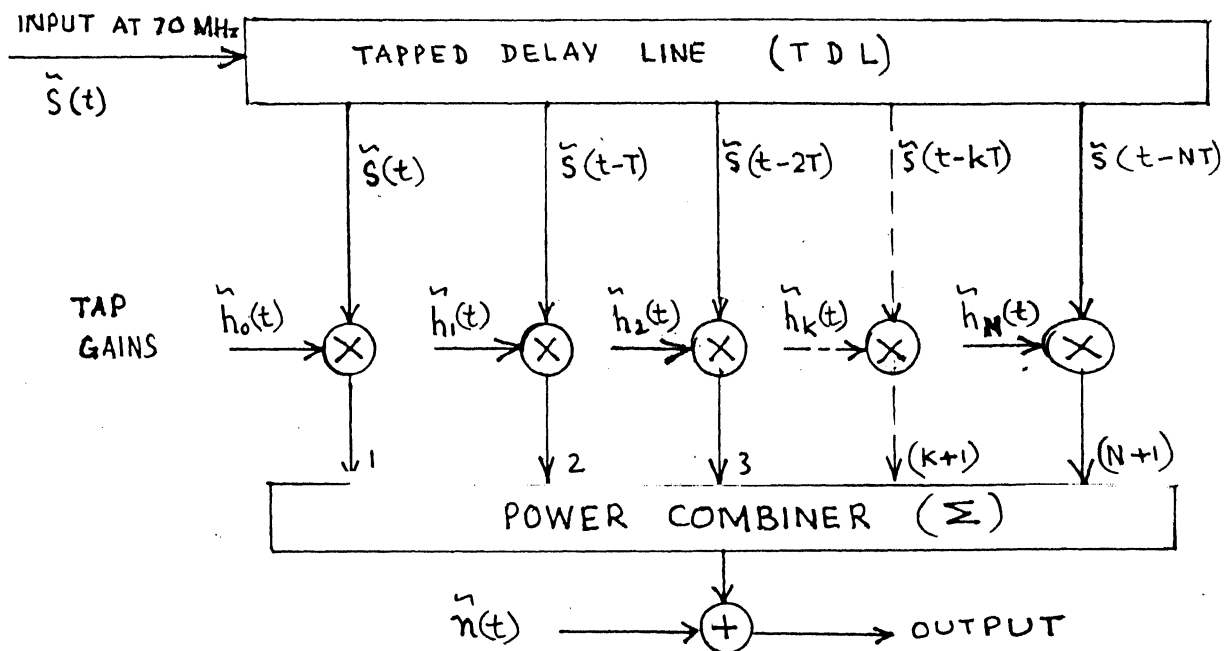


Fig. 1.3.3: A Tapped Delay Line Model.

Define

$$\tilde{h}_k(t) = h_{kc}(t) + jh_{ks}(t), \quad k = 0, 1, \dots, N \quad (1.3.7)$$

where  $h_{kc}(t)$  and  $h_{ks}(t)$  are the inphase and quadrature components of the channel gain variations at the  $k$ th tap. From the WSSUS model, each  $h_{kc}(t)$  and  $h_{ks}(t)$  are assumed to be real Gaussian noise with spectrum as given by the corresponding scattering function  $S(f, \tau)$  for  $\tau = kT$ .

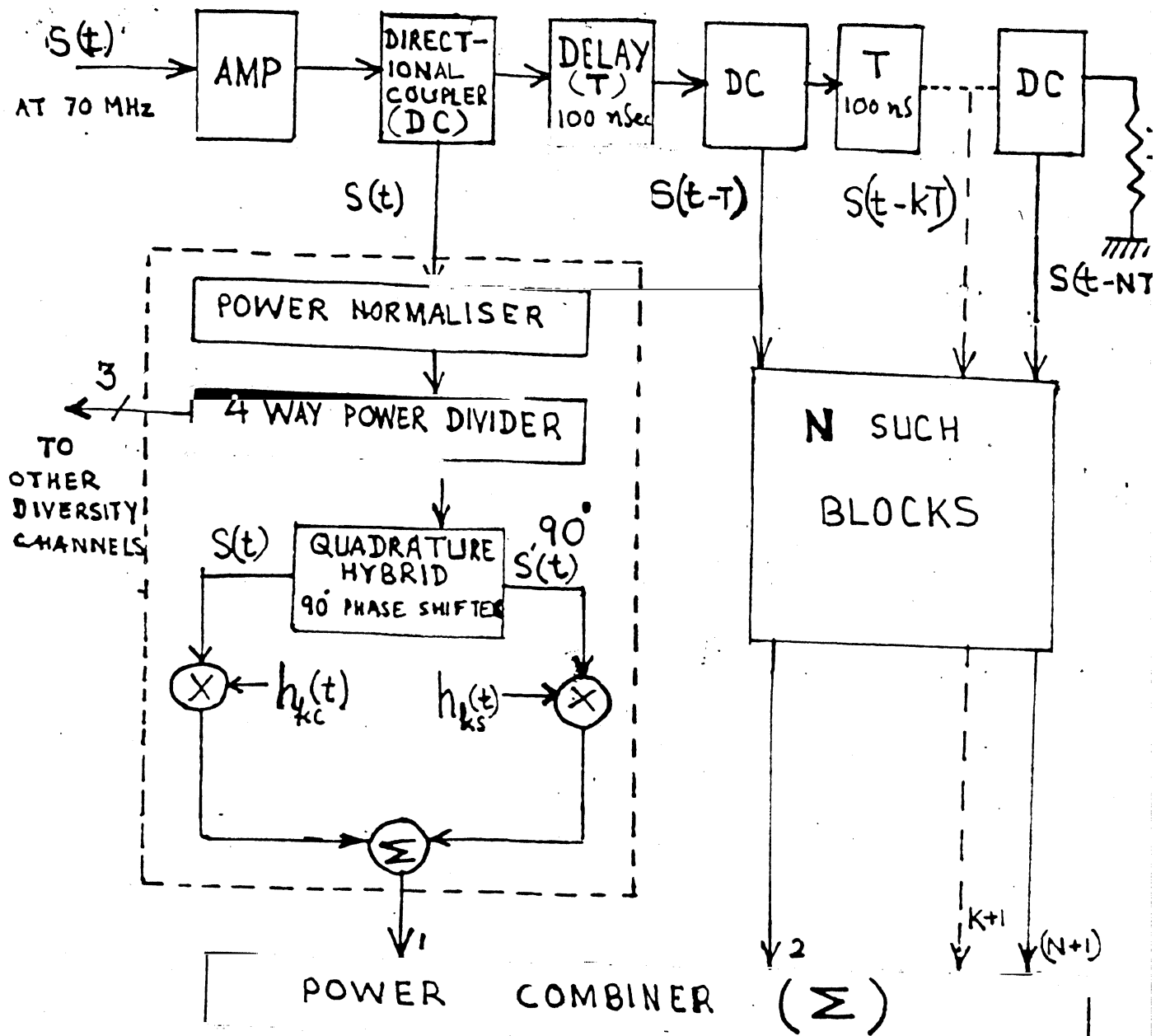
Thus, a major problem in the simulator is how to synthesize the  $(N+1)$  complex Gaussian noise sources as described in Eqn. (1.3.7) with controlled spectrum to provide the desired scattering function of the channel model. A hardware method of implementing

the  $2(N+1)$  synthesized programmable Gaussian noise sources using modern microprocessors and LSI devices is the main thrust of this work.

#### 1.4 BRIEF DESCRIPTION OF THE SIMULATOR:

The practical implementation of the Tapped delay line model shown in Fig. 1.3.3 is best done at the standard intermediate frequency (70 MHz) of the Tropo Communication Systems. A detailed block diagram of the Tropo-channel simulator [ 9] is shown in Fig. 1.4.1.

The input (at 70 MHz IF) to the simulator undergoes time delay of 100 nsec. at each tap. Directional couplers are used for tapping the signal and the power from each tap is normalised to the desired level. A four way power divider is used to provide power for simulating four diversity channels. As shown in Fig. 1.4.1  $S'(t)$  is derived from  $S(t)$  by a quadrature hybrid ( $90^\circ$  phase shifter) to obtain the cosine and sine components.  $h_{kc}(t)$  and  $h_{ks}(t)$  are the real Gaussian noises synthesised in the random noise generator. These modulate the  $S(t)$  and  $S'(t)$  components to provide the desired tap gain. Signals from all the taps are combined using a power combiner and white Gaussian noise is added to produce the desired power spectrum.



OUTPUT

FOR

DIVERSITY CHANNEL 1

Fig. 1.4.1: Block Diagram of the Tropo-channel Simulator



### 1.5 THESIS OUTLINE:

Various methods available for random noise generation and the method adopted in this work are discussed in the next chapter. The hardware implementation is covered in Chapter 3 followed by a detailed discussion of the software in Chapter 4. Finally conclusion and suggestions for further improvement are covered in Chapter 5.

## CHAPTER 2

### THE ALGORITHM OF THE NOISE GENERATOR

In this chapter, the different methods of generating Normal variates are reviewed. The method adopted in this work is discussed in further detail. The procedures which may be used to generate many independent sample sequences at fast enough rate and those with which their spectral characteristics can be controlled through programmable filters are also described.

#### 2.1 GENERATION OF NORMAL VARIATES:

There are basically two approaches to the generation of Normal Variates. The first one being the Central Limit Theorem (C.L.T.) approach. The second approach is to generate deviates in the interval  $[0,1]$  and to apply a suitable transformation to obtain the Normal distribution with the desired mean and variance denoted as  $N(\text{mean}, \text{variance})$ . These approaches are the subject of discussion in this section.

##### 2.1.1 Central Limit Theorem Approach:

The C.L.T., states that if  $X_1, X_2, \dots, X_n$  be a set of zero mean random variables which are statistically independent, all having the same probability distribution function

with identical variance  $\sigma_x^2$  for each  $X_k$ , then the random variable

$$Y = \sum_{k=1}^n X_k \quad (2.1.1)$$

approaches a Gaussian distributed random variable with zero mean and variance  $n\sigma_x^2$  for sufficiently large  $n$ .

It is important to note that the C.L.T., gives only the 'limiting' form of the probability distribution function of the random variable  $Y$  as  $n$  tends to infinity. When  $n$  is finite, the Gaussian approximation may be relatively poor.

A typical example of the C.L.T., implemented in the simplest form is the Random walk. The random walk experiment consists of  $n$  trials of generating at each trial a  $+1$  or  $-1$ , with equal probability and incrementing or decrementing a counter (initialised to zero) by the number obtained at each trial, and reading the final count obtained. For example if  $n = 1000$ , the resulting count is a sample of integers from  $-1000$  to  $+1000$  approximating the distribution  $N(0,1000)$ . This is the method used in [ 8 ].

### 2.1.2 Normal Variates from Uniform Deviates:

There are two methods used to generate Normal Variates from uniform deviates. In both these methods uniform deviates

in  $[0,1]$  are first generated and then a suitable transformation is applied to generate variates with the desired distribution.

First of these methods is known as the inverse transformation method. To understand this method mathematically consider a continuous random variable  $X$  with  $f_X(x)$  and  $F_X(x)$ , as the desired density and distribution functions.

$$F_X(x) = \int_{-\infty}^x f_X(t) dt \quad (2.1.2)$$

Now consider the new random variable

$$U = F_X(x) \quad (2.1.3)$$

Then  $U$  has nonzero probability in  $(0,1)$  since  $F_X(x)$  can assume values in  $(0,1)$  only. The density function of  $U$  is

$$f_U(u) = |J| f_X(x) \quad (2.1.4)$$

where  $|J|$  is the determinant of the Jacobian of transformation and  $J = \frac{\partial x}{\partial u}$ . In the present case

$$J = \frac{1}{f_X(x)} \quad (2.1.5)$$

and

$$f_U(u) = \begin{cases} 1 & 0 \leq u \leq 1 \\ 0 & \text{else-where,} \end{cases}$$

which is the density function of a uniformly distributed random variable on (0,1). Taking inverse transformation of (2.1.3)

$$X = \phi[F_X(x)] = \phi(U) \quad (2.1.6)$$

This would generate X with any desired probability distribution [2 ].

In the second method using Uniform deviates to generate Normal variates, two independent Normal variates

$$\begin{aligned} X_1 &= (-2 \log U_1)^{1/2} \cos (2\pi U_2) \\ X_2 &= (-2 \log U_2)^{1/2} \sin (2\pi U_2) \end{aligned} \quad (2.1.7)$$

with  $N(0,1)$  can be generated provided  $U_1$  and  $U_2$  are independent deviates from  $U(0,1)$  [4 ].

Medium and large computers have good computational facilities and mathematical functions such as log, square root, sine and cosine are easily performed. As such, the above method is popularly used in computer simulation.

### 2.1.3 On Microcomputer Implementation:

The method of (2.1.7) is computationally intensive and therefore unsuited for implementation on microcomputers. While the C.L.T., approach is very simple and attractive, it can be rather slow (considering typically 1000 Uniformly

distributed samples for every <sup>single</sup>  $\angle$  Normally distributed sample) unless additional high speed hardware is used.

A third alternative is to generate uniform variates at a fast enough rate and then using them to address a look up table in which the inverse distribution function of the Normal variates is stored. This is the approach taken in this work. Chapters 3 and 4 give a more detailed description of this method.

## 2.2 GENERATION OF UNIFORM DEVIATES:

The starting point in the generation of Normal variates is to generate Uniform deviates first. In the present section a brief review of various methods available for Uniform number generation is carried out.

### 2.2.1 Considerations:

There are three basic considerations which play important role in determining whether or not a particular source provides random numbers that are adequate for experimental purposes. The numbers must pass statistical tests [3] designed to reveal departures from independence and uniformity.

Secondly, if the uniformly distributed variates  $x_i$  are to be represented by integers, (as is the case in digital representation) say  $0 \leq x_i \leq M$ , then  $M$  should be

sufficiently large to approximate the continuous  $u(0,1)$  well.

Thirdly the efficiency with which a particular source produces random numbers must be very good for real time random number generators. The faster an algorithm produces a random number, under the constraints of the microcomputer architecture, the more desirable that algorithm is.

The presence of sufficient independence and uniformity to preserve the integrity of the particular experiment under consideration is the most important criterion in determining the adequacy of the system. Keeping this in mind a balance may be struck in achieving the other properties desired in the random number generation process.

#### 2.2.2 Table Look Up Method:

In this method for generating uniform random numbers a table of random numbers is stored in the computer memory and accessed at a certain rate. Each number is a random integer in the set  $[0,1,2,\dots,2^n-1]$ , so that theoretically the probability of drawing a particular integer is  $1/2^n$ . Present day storage capabilities and faster access times makes this method attractive for implementation. However, it is to be ensured that the data stored in the look up table has the requisite statistical properties.

### 2.2.3 Pseudo Random Number Generator:

A more popular approach in the generation of uniform deviates is the Pseudo Random Number Generator. It produces a non-random sequence of numbers, each being completely determined by its predecessor, and consequently all numbers being determined by the initial numbers. However if the parameters of the generator are chosen in an informed way, the numbers appear sufficiently independent and uniform for most practical purposes [2].

A Pseudo random bit sequence generator (PRBSG) is a feed back shift register circuit, with feed-back carefully chosen to produce maximal length sequence of  $(2^n - 1)$  bits, where  $n$  is the number of shift registers used. Every feed-back shift register circuit is described by the polynomial relation

$$x^n = a_{n-1} x^{n-1} \oplus a_{n-2} x^{n-2} \oplus \dots \oplus a_1 x \oplus a_0$$

or equivalently by  $P(x) = 0$ , where

$$P(x) = x^n \oplus a_{n-1} x^{n-1} \oplus a_{n-2} x^{n-2} \oplus \dots \oplus a_1 x \oplus a_0 \quad (2.2.1)$$

In Eqn. (2.2.1) the coefficients  $a_k$  are either 0 or 1 and the symbol  $\oplus$  represents the modulo-2 summation. For example, the feedback shift register, corresponding to the polynomial  $x^4 \oplus x \oplus 1$ , shown in Figure 2.2.1.



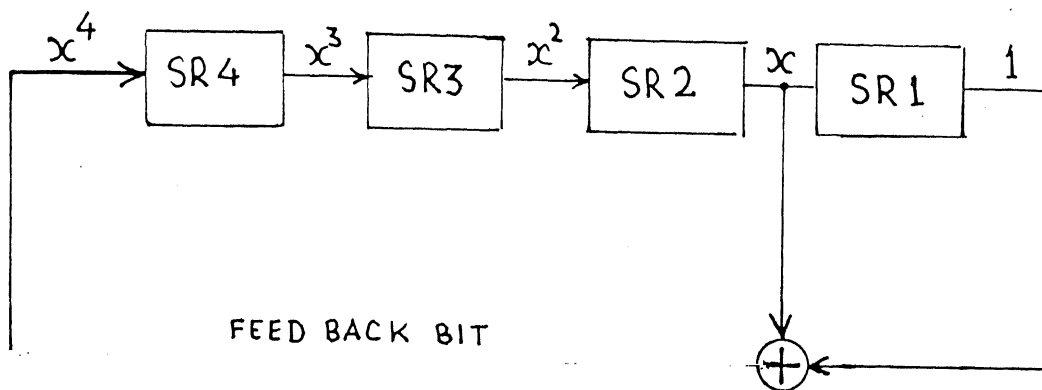


Fig. 2.2.1: Example of Feed Back Shift Register.

It is well known from coding theory [6] that only primitive polynomials implemented in feed back shift registers produce the desired maximal length sequences. In this work we have chosen the following primitive polynomial [7], for PRBSG implementation,

$$P(x) = x^{34} \oplus x^{27} \oplus x^2 \oplus x \oplus 1 \quad (2.2.2)$$

This produces a sequence length of  $(2^{34}-1)$  bits.

If eight parallel polynomials of the form of (2.2.2), initialised to different states sufficiently apart from one another in the maximal length sequence, to ensure statistical independence, are made to produce eight parallel bit streams then a set of eight parallel bits at any instant of time could form a random number from 0 to  $2^8 (=255)$  with uniform probability of occurrence  $\frac{1}{2^8+1} (= \frac{1}{256})$ .

To ensure that each of the eight PRBSGs are initialised to different far apart states, it was decided to run a single PRBSG and record it's state after every  $2^{31}$  shifts. Eight such recordings would thus complete one maximal length sequence run of  $(2^{34}-1)$  and give us 34 bytes which would be the initial value for each of the eight PRBSGs.

With this in mind a software approach was tried using 8085 assembly language program of Appx. 'A'. This was very slow (takes 34 hours to get the first reading) and hence abandoned. A hardware as shown in Fig. 2.2.2 was fabricated. Assembly language program of Appx. 'B' was used to stop the PRBSG under interrupt control after every  $(2^{31}-2^{15})$  shifts counted by programmable counters (8253). By this method the eight states were recorded as per Table 1. The hardware was sufficiently fast and a complete run of the maximal length sequence of  $(2^{34}-1)$  bits was over in two hours.

Thus starting with the initial state of Table 1 the eight PRBSGs would generate eight bit words (0 to 255) with uniform probability distribution in (0,1). Corresponding to each 8 bit address, random data, (calculated by inverse transformation method) can be accessed. The calculation of the random numbers with  $N(0,2304)$  is explained in the next section.

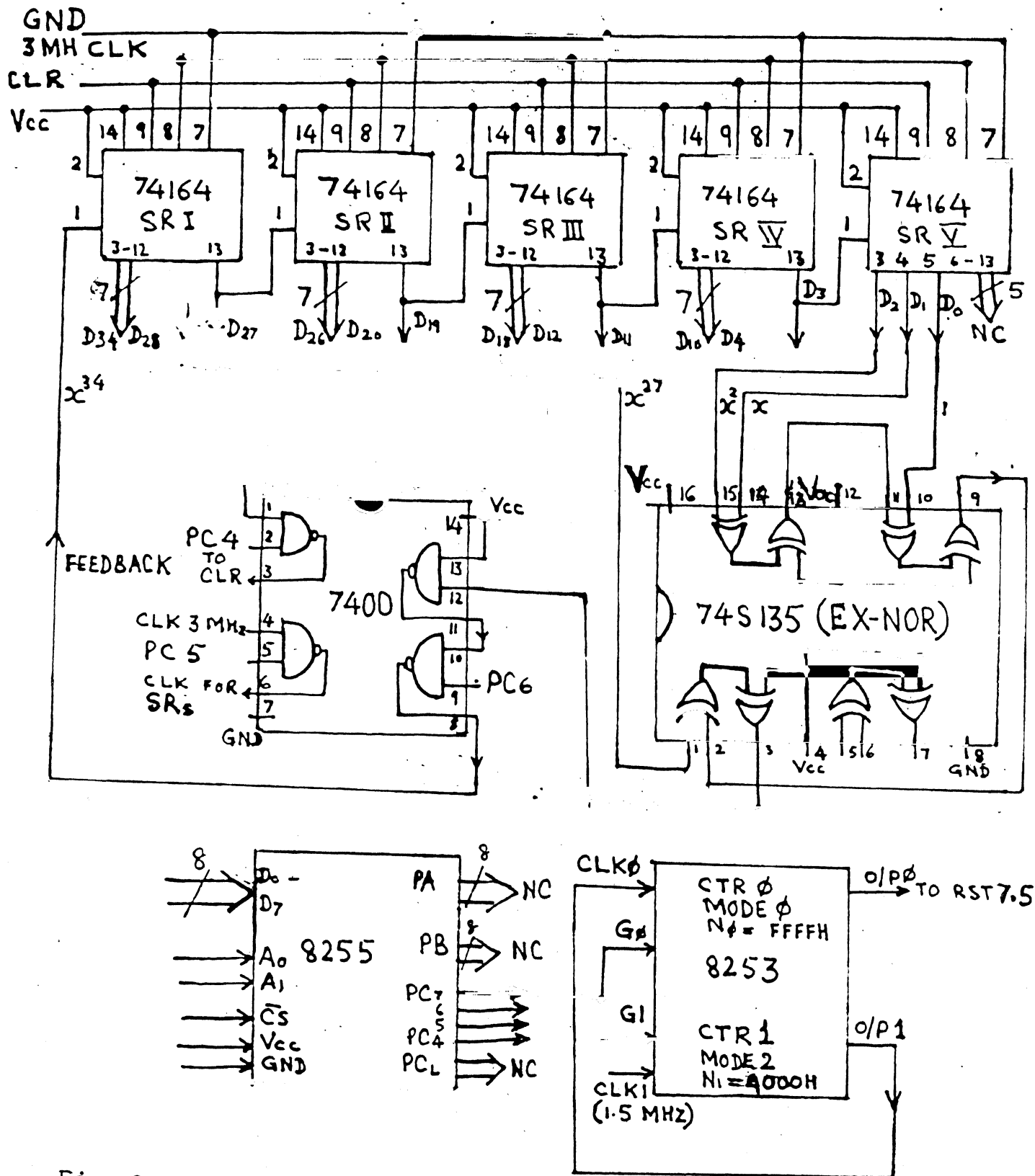


Fig. 2.2.2: An implementation of a 34 Bit PRBSG under interrupt control.

Table 1: State of Shift Registers after every ( $2^{31}-2^{15}$ ) Shifts

Coefficients of	Initial	1st	2nd	3rd	4th	5th	6th	7th	HEX Equiv- alent
	Readings								
$x^{33}$	1	0	0	0	0	0	0	0	80
$x^{32}$	1	1	0	1	0	0	0	0	D0
$x^{31}$	1	1	0	0	0	1	0	0	C4
$x^{30}$	1	0	1	0	0	1	0	1	A5
$x^{29}$	1	1	0	0	1	1	0	1	CD
$x^{28}$	1	0	0	0	1	0	0	0	88
$x^{27}$	1	0	1	1	0	0	1	1	B3
$x^{26}$	1	0	0	1	0	1	1	0	96
$x^{25}$	1	0	0	0	0	1	1	0	86
$x^{24}$	1	0	0	1	1	0	1	0	9A
$x^{23}$	1	1	0	0	0	0	1	0	C2
$x^{22}$	1	0	1	0	0	0	1	0	A2
$x^{21}$	1	1	1	0	1	1	1	0	EE
$x^{20}$	1	0	0	0	0	1	0	0	84
$x^{19}$	1	0	1	1	0	0	0	1	B1
$x^{18}$	1	1	0	0	0	0	0	1	C1
$x^{17}$	1	0	0	0	1	1	1	1	8F

Contd....

Coefficients of	Initial	1st	2nd	3rd	4th	5th	6th	7th	HEX Equiva- lent
				Readings					
$x^{16}$	1	0	0	0	0	1	1	0	86
$x^{15}$	1	0	0	0	1	0	0	0	88
$x^{14}$	1	0	0	0	0	0	0	0	80
$x^{13}$	1	1	0	0	0	0	0	0	C0
$x^{12}$	1	0	0	0	0	1	0	0	84
$x^{11}$	1	0	0	0	1	1	0	0	8C
$x^{10}$	1	1	0	0	1	1	0	0	CC
$x^9$	1	0	0	0	1	0	0	0	88
$x^8$	1	0	0	1	0	0	0	0	90
$x^7$	1	0	0	0	0	0	1	0	82
$x^6$	1	0	0	0	1	0	1	0	8A
$x^5$	1	0	0	1	0	1	0	0	94
$x^4$	1	0	0	0	1	0	0	0	88
$x^3$	1	0	1	0	0	1	1	0	A6
$x^2$	1	0	0	0	1	1	0	0	8C
$x^1$	1	1	0	0	0	0	1	0	C2
$x^0$	1	0	1	0	0	0	0	0	A0

### 2.3 CALCULATION OF RANDOM NUMBERS:

For calculating the random numbers with Normal distribution we follow the inverse transformation method explained in Section 2.1.2. Consider the probability density function (p.d.f.)

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{x^2}{2\sigma^2}\right) \quad (2.3.1)$$

of a Normal random variable  $X$  with  $N(0, \sigma^2)$  as shown in Figure 2.3.1.

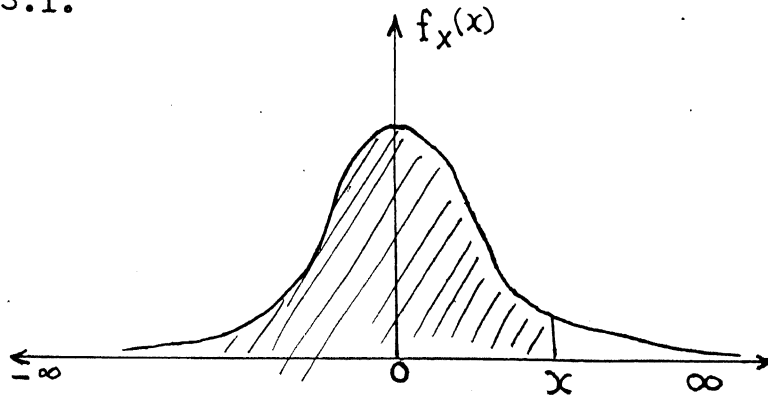


Fig. 2.3.1: The Normal p.d.f. curve.

For  $\sigma^2 = 1$ , the shaded area gives the error function

$$\text{erf}(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^x \exp\left(-\frac{t^2}{2}\right) dt \quad (2.3.2)$$

This also represents the distribution function  $F_X(x)$  of the random variable  $X$ .  $F_X(x)$  is plotted in Figure 2.3.2.

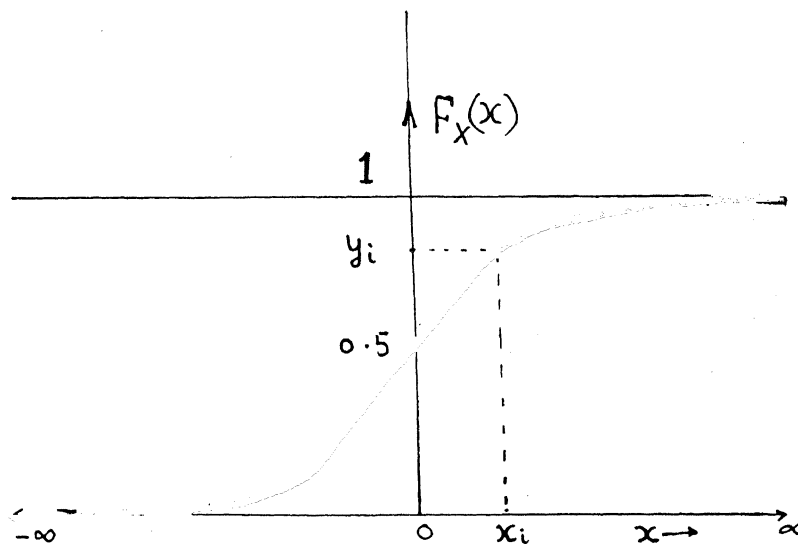


Fig. 2.3.2: Normal Distribution Function Plot.

Reference [5] tabulates the error function values (from 0.5 to 1.0) and the corresponding values of  $X$  (from 0 to 4).  $[1-\text{erf}(x)]$  gives the corresponding values for  $x < 0$ . Thus knowing  $F_X(x)$  we can find the value of  $X$  from [5].

By dividing numbers from 0 to 255 by 256 we get 256 points  $y_i$  in  $[0,1]$  on the vertical axis as shown in Figure 2.3.2. Corresponding to each such point  $y_i$  we get one point  $x_i$  from [5].

For the highest number 255, the ratio  $y_i = 255/256 = 0.9960$  corresponds to  $x_i = 2.65$ . Thus with 8 bit words we could possibly achieve  $2.65\sigma$ . Larger range of output requires greater word lengths. With 8 bits we could represent numbers from  $[-127, 127]$  in sign magnitude representation. Equating the end values  $2.65\sigma = 127$ , we get  $\sigma = 48$ . Thus, if all  $x_i$  are scaled by 48 we get the full range of random numbers from -127 to 127.

For example corresponding to address location 200,  
 $y_{200} = \frac{200}{256} = 0.7813$ . For this value of  $\text{erf}(x)$  we get from [5]

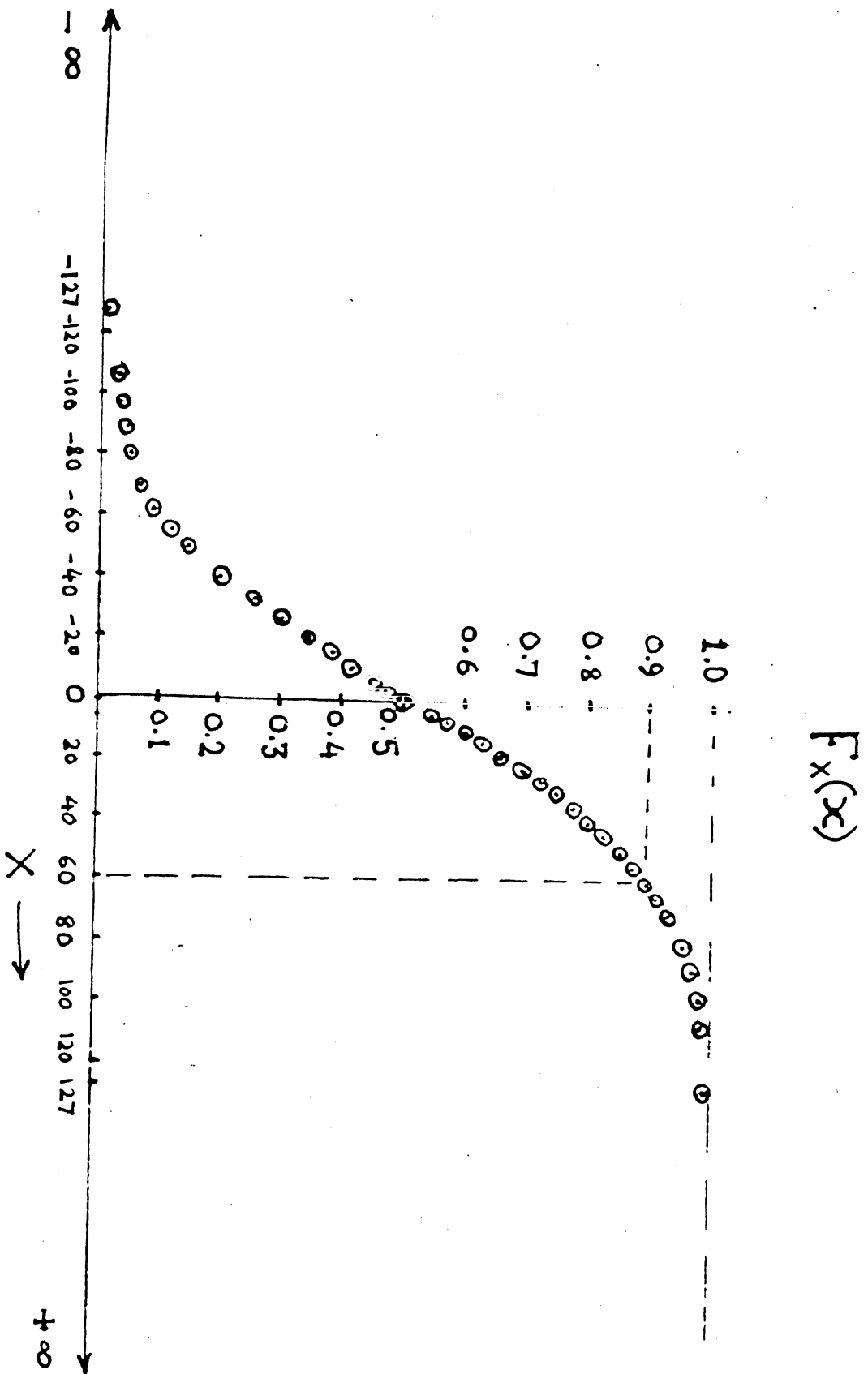


Fig. 2.3.3: A plot of Normal random numbers of Table 2.



the corresponding value of  $x_{200} = 0.775$ . Scaling this random number by 48 we get the actual random number 37 which would form the contents of memory location 200 in the look up table. The random numbers corresponding to address location from 128 to 255 were calculated and tabulated in Table 2. For address locations 0 to 127 the random numbers are a mirror image, hence we take exactly the same values with a negative sign. Their plot shown in Figure 2.3.3 resembles the plot of  $F_X(x)$  of Figure 2.3.2.

The microcomputer implementation of the PRBSGs and the table look up procedure is further discussed in Chapters 3 and 4.

#### 2.4 SPECTRAL SHAPING OF THE RANDOM NOISE:

The random noise samples from the output of the dedicated random number generator could be distributed to digital to analog converters (DACs) to generate the required complex processes  $h_{kc}(t)$  and  $h_{ks}(t)$  defined in (1.3.7). However these will produce symmetric power spectrum only [9]. In actual fact these are known to possess asymmetric power spectrum as well [9]. Examples of power spectra are shown in Figure 2.4.1.

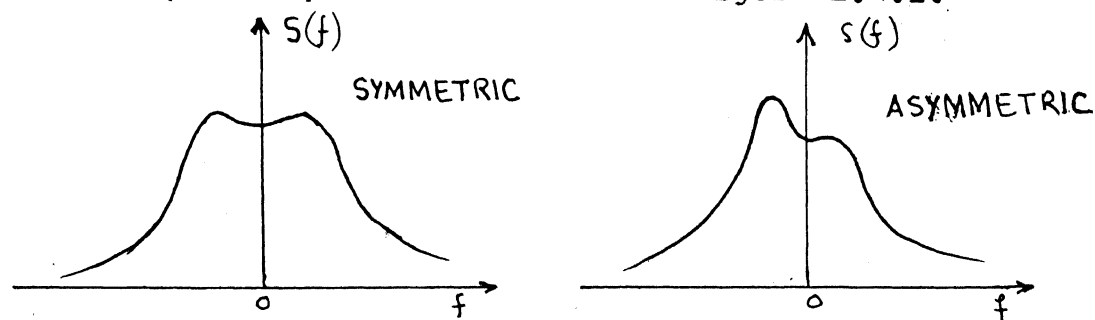


Fig. 2.4.1: Examples of Power Spectra.

Table 2: Random Number (R.N.) with N (0,2304)

Sl. No.	Ratio $y_i$	$x_i \times 48$ DEC	RN HEX	Sl. No.	Ratio $y_i$	$x_i \times 48$ DEC	RN HEX
128	.5	0	00	158	.6172	14	0E
129	.5039	0	00	159	.6211	14.8	0F
130	.5078	01	01	160	.625	15.3	0F
131	.5117	01	01	161	.6289	15.8	10
132	.5156	02	02	162	.6328	16.2	10
133	0.5195	02	02	163	.6367	16.8	11
134	.5234	03	03	164	.6406	17.3	11
135	.5273	03	03	165	.6445	17.7	12
136	.5313	04	04	166	.6484	18.3	12
137	.5352	04	04	167	.6523	18.8	13
138	.5391	05	05	168	.6563	19.3	13
139	.5430	05	05	169	.6602	19.8	14
140	.5469	06	06	170	.6641	20.4	14
141	.5508	06	06	171	.6680	20.8	15
142	.5547	07	07	172	.6719	21.3	15
143	.5586	07	07	173	.6758	21.8	16
144	.5625	08	08	174	.6797	22.4	16
145	.5664	08	08	175	.6836	22.9	17
146	.5703	09	09	176	.6875	23.5	17
147	.5742	09	09	177	.6914	24	18
148	.5781	09	09	178	.6953	24.5	18
149	.5820	10	0A	179	.6992	25.1	19
150	.5859	10	0A	180	.7031	25.7	1A
151	.5898	11	0B	181	.7070	26.3	1A
152	.5938	11	0B	182	.7109	26.6	1B
153	.5977	12	0C	183	.7148	27.1	1B
154	.6016	12	0C	184	.7188	27.7	1C
155	.6055	13	0D	185	.7227	28.2	1C
156	.6094	13	0D	186	.7266	28.9	1D
157	.6133	14	0E	187	.7305	29.4	1D

contd...

Sl. No.	Ratio $y_1$	xi x 48 DEC	RN HEX	Sl. No.	Ratio $y_1$	xi. x 48 DEC	RN HEX
188	.7344	30.1	1E	217	.8477	49.3	31
189	.7383	30.7	1F	218	.8516	50.0	32
190	.7422	31.2	1F	219	.8555	50.9	33
191	.7461	31.8	20	220	.8594	51.6	34
192	.7500	32.4	20	221	.8633	52.6	35
193	.7539	32.9	21	222	.8672	53.4	35
194	.7578	33.6	22	223	.8711	54.3	36
195	.7616	34.2	22	224	.8750	55.2	37
196	.7656	34.9	23	225	.8789	56.2	38
197	.7695	35.4	23	226	.8828	57.1	39
198	.7734	36.0	24	227	.8867	58.0	3A
199	.7773	36.9	25	228	.8906	59.0	3B
200	.7813	37.2	25	229	.8945	60.1	3C
201	.7852	37.9	26	230	.8984	61.0	3D
202	.7891	38.6	27	231	.9023	62.2	3E
203	.7930	39.2	27	232	.9063	63.3	3F
204	.7969	39.9	28	233	.9102	64.4	40
205	.8008	40.4	28	234	.9141	65.6	42
206	.8047	41.1	29	235	.9180	66.8	43
207	.8086	41.9	2A	236	.9219	68.1	44
208	.8125	42.6	2B	237	.9258	69.2	45
209	.8164	43.3	2B	238	.9297	70.6	47
210	.8202	44	2C	239	.9336	72.1	48
211	.8242	44.7	2D	240	.9375	73.4	49
212	.8281	45.6	25	241	.9414	75.2	4B
213	.8320	46.2	2E	242	.9453	76.8	4D
214	.8359	46.8	2F	243	.9492	78.7	4F
215	.8398	47.7	30	244	.9531	80.6	51
216	.8438	48.5	30	245	.9570	82.6	53

contd...

Sl. No.	Ratio $y_i$	$x_i \times 48$ DEC	RN HEX	Sl. No.	Ratio $y_i$	$x_i \times 48$ DEC	RN HEX
246	.9609	84.5	55	251	.9805	99.4	63
247	.9648	86.9	57	252	.9844	103.2	67
248	.9688	89.3	59	253	.9883	108.8	6D
249	.9727	92.2	5C	254	.9922	116.2	74
250	.9766	95.5	60	255	.9960	127.2	7F

For spectral shaping of these processes, therefore, we need some mechanism with programmable parameters. Digital filters are ideally suited for the purpose. For example a first order digital filter as shown in Figure 2.4.2.

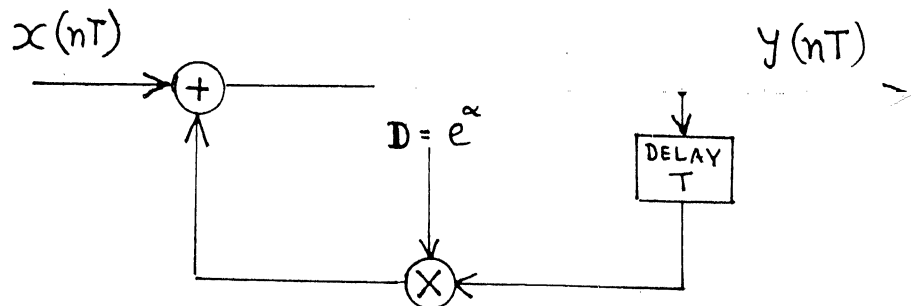


Fig. 2.4.2: First Order Digital Filter.

The output of this filter is

$$y(nT) = x(nT) + Dy(nT-T) \quad (2.4.1)$$

where  $T$  is the sampling period and  $n$  is an integer including 0.  $D = e^{\alpha}$  is the feed back gain. The response of the filter to a digital impulse (1000...0) is

$$y(nT) = D^n \quad (2.4.2)$$

$D < 1$

This is stable provided  $|D| < 1$ . The impulse response of an equivalent single pole analog filter with bandwidth  $f_0$  Hzs is

$$y(t) = e^{-2\pi f_0 t} \quad (2.4.3)$$

or equivalently at instant  $t = nT$

$$y(nT) = e^{-2\pi f_0 nT} \quad (2.4.4)$$

Equating (2.4.2) and (2.4.4)

$$D = e^{-2\pi f_o T} \quad (2.4.5)$$

For a given  $T$  and  $f_o$ ,  $D$  has a particular value given by (2.4.5). If we keep both parameters  $f_o$  and  $T$  programmable then we would be required to calculate the value of  $D$  depending on what  $f_o$  and  $T$  the user wants on a particular tap. However calculation would be time consuming on microcomputer. Hence a look up table was prepared. Corresponding to  $f_o$  from 1 to 16 Hzs and  $T$  from 1 to 16 milliseconds the values of  $D$  were calculated and tabulated in Table 3.

In the next chapter the aspects governing the hardware implementation are highlighted while explaining the hardware used for this work.

Table 3: Coefficients (D) for the First Order Digital Filter of Fig. 2.4.2

BW(Hzs) ROM ADDR		Sampling Time (msec)															
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	6F00	7E	7E	7E	7D	7C	7B	7A	7A	79	78	77	77	76	75	74	74
2	6F10	7E	7D	7B	7A	78	77	75	74	72	71	70	6E	6D	6B	6A	69
3	6F20	7E	7B	7A	77	75	72	70	6E	6C	6A	68	66	64	62	60	5F
4	6F30	7D	7A	77	74	71	6E	6B	69	66	64	61	5F	5D	5A	58	56
5	6F40	7C	78	75	71	6E	6A	67	64	61	5E	5B	58	55	53	50	4E
6	6F50	7B	77	72	6E	6A	66	62	5E	5B	57	54	51	4E	4B	48	45
7	6F60	7B	75	70	6C	67	63	5E	5A	57	53	4F	4C	49	46	43	40
8	6F70	7A	74	6E	69	64	5E	5A	56	52	4E	4A	46	43	3F	3D	39
9	6F80	79	73	6D	66	61	5C	57	52	4E	49	45	42	3E	3B	38	35
10	6F90	78	71	6A	64	5E	58	53	4E	49	44	40	3C	39	35	32	2F
11	6FA0	77	6F	67	60	59	54	4E	48	43	3F	3A	36	33	2F	2C	29
12	6FB0	77	6E	66	5E	57	51	4B	45	40	3B	37	33	2F	2C	28	25
13	6FC0	76	6C	63	5C	54	4E	47	42	3D	38	33	2F	2B	28	25	22
14	6FD0	75	6B	62	59	51	4A	44	3E	39	34	2F	2B	27	24	21	1E
15	6FE0	75	6B	62	59	51	4A	44	3E	39	34	2F	2B	27	24	21	1E
16	6FF0	74	69	5F	56	4E	46	40	39	34	2F	2B	27	23	1F	1D	1B

## CHAPTER 3

### HARDWARE IMPLEMENTATION

In this chapter, a microcomputer workstation based implementation of the noise generator using the algorithm discussed in Sections 2.3 and 2.4 is presented.

#### 3.1 THE NOISE GENERATOR:

The noise sources  $h_{kc}(t)$  and  $h_{ks}(t)$  shown in Figure 2.4.1, used as tap gains for the complex multipliers, can be synthesised by the scheme shown in Figure 3.1.1.

The Basic White Gaussian Noise Generator card generates the random noise samples. These samples are provided to the master processor under interrupt control for further processing. The master processor performs digital filtering and distributes the coloured noise samples to the digital to analog converters (DACs) which form the output card. The outputs

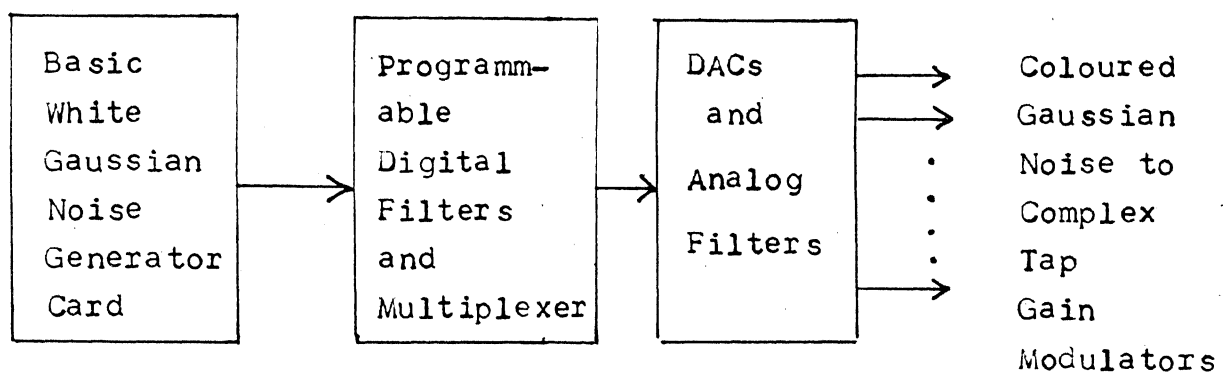


Fig. 3.1.1: Block Diagram of the Noise Generator



from the DACs are shaped by analog filters before finally presenting them as  $h_{kc}(t)$  and  $h_{ks}(t)$  to the main simulator of Figure 2.4.1. These blocks shown in Figure 3.1.1 are further elaborated in the sections to follow.

### 3.2 THE WHITE GAUSSIAN NOISE GENERATOR CARD:

This portion of the hardware forms the heart of the main simulator of Figure 2.4.1. A single chip microcomputer 8741A provides the white noise samples with  $N(0,2304)$ . The samples are generated every 250  $\mu\text{sec}$ , thus providing 4000 samples every seconds. These have symmetric power spectrum, and as discussed in Section 2.4, there is a need for the spectral Shaping of these samples to produce the desired asymmetry in the power spectrum. However digital filtering needs fast multiplication and accumulation. Hence we decided to use TDC 1008 multiplier accumulator chip for this purpose. The block diagram of the white Gaussian Noise Generator Card is shown in Figure 3.2.1. For program storage one EPROM is included in this card. Control signals and chip select signals are provided by the decoder logic. These hardware blocks are discussed below in further detail.

#### 3.2.1 The Single Chip Microcomputer (8741A):

8741A is used in slave mode to generate the white Gaussian noise samples. These samples are read by master

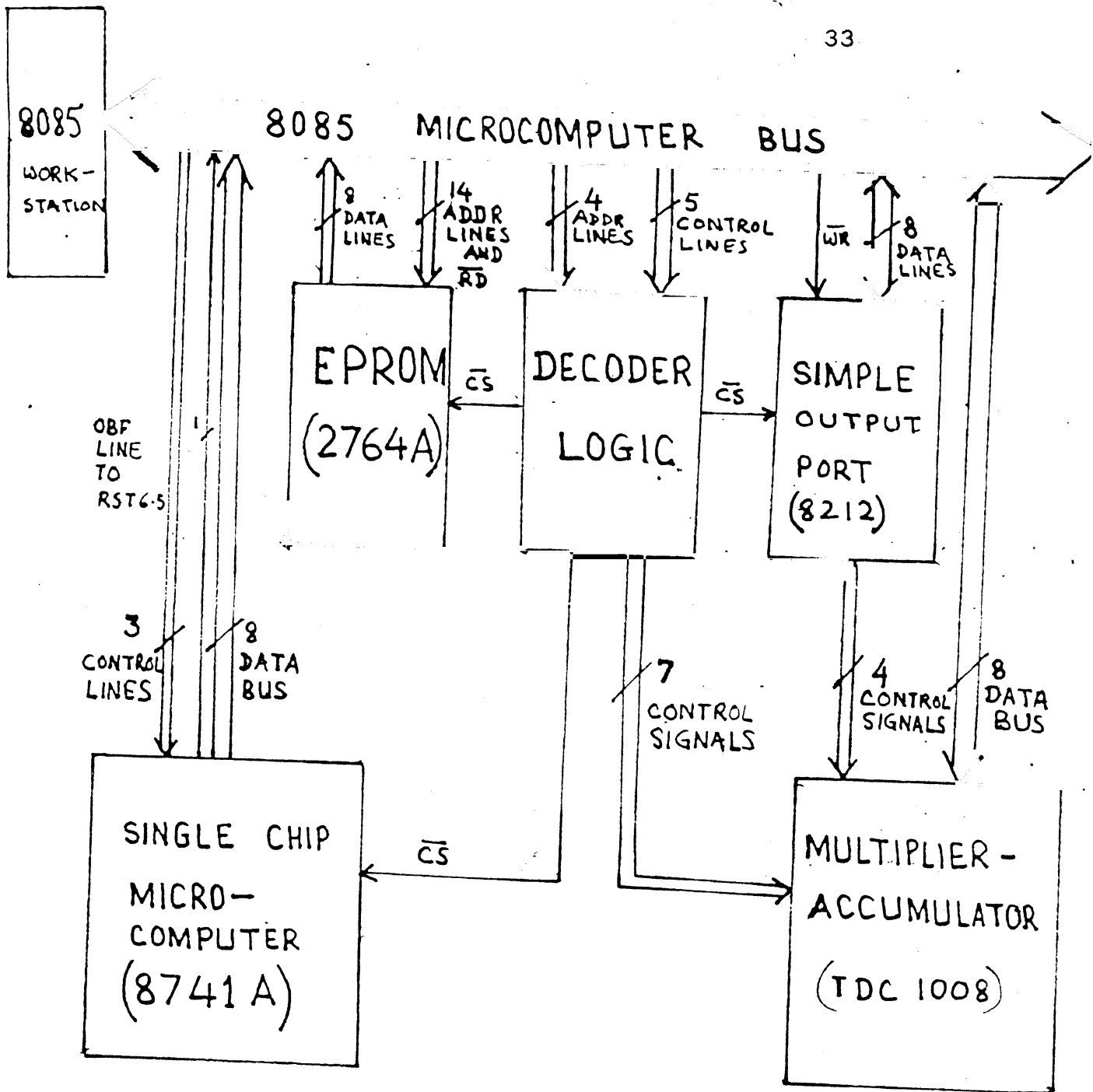


Fig. 3.2.1: Block diagram of the white Gaussian noise generator card.

processor 8085 under interrupt control. It will not be out of place to review the buffer structure of 8741A to understand how this is done. 8741A buffer structure is shown in Figure 3.2.2.

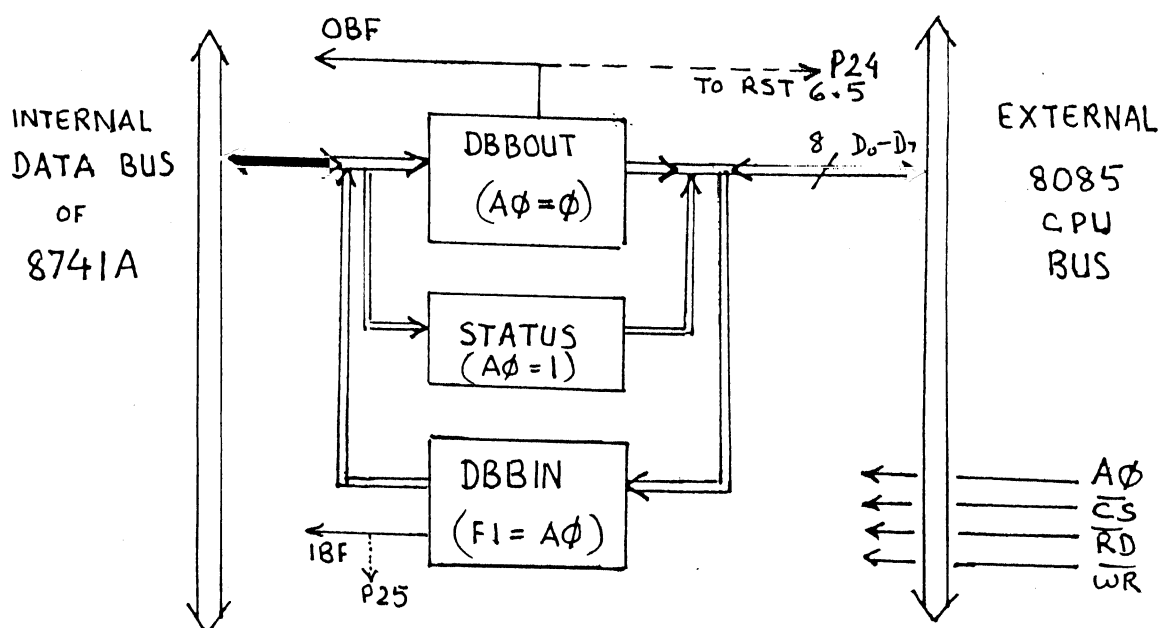


Fig. 3.2.2: The Buffer Structure of 8741A

8741A has two registers DBBOUT and DBBIN. STATUS register is derived from the DBBOUT register. DBBOUT register becomes full when 8741A has written data into it. It becomes empty when the external processor has read from DBBOUT register. When a 1 is written to output line P24 of port 2 and when EN FLAGS instruction has been executed, Output Buffer Full (OBF) signal is available on P24 (pin 35 of 8741) for hand shake with external CPU. OBF is connected to

interrupt line RST 6.5 of 8085 (in the present work ). When data is written to DBBOUT by 8741A, OBF goes high. It goes low only when this data has been read by external CPU. 8741A also has JOBF (Addr ) instruction to continuously check if the OBF is high or low. Accordingly the 8741A CPU branches to the address (Addr ) or continues sequentially to work out the next sample and put it out on to the DBBOUT register. Table 4 gives the truth table for read/write operations of 8741A.

Table 4: Read/Write Operations of 8741A

Operation	$\overline{CS}$	$\overline{RD}$	$\overline{WR}$	AO
Read DBBOUT	0	0	1	0
Read STATUS	0	0	1	1
Write DBBIN (data)	0	1	0	0
Write DBBIN (command)	0	1	0	1

DBBIN and STATUS registers of 8741A are not used in the present work, hence  $\overline{WR}$  of 8741A is permanently connected to  $V_{cc}$ . To read the DBBOUT of 8741A we must ensure  $\overline{CS}$ ,  $\overline{RD}$  and AO to be 0.

8741A is a 40 pin IC. It's pinout is shown in Appendix 'D'. Ports 1 and 2 are not used. Only P24 line of port 2 is used as OBF line to interrupt 8085.  $\overline{\text{RESET}}$  is an active low input which resets 8741A and program execution starts from address 000H. An active low on  $\overline{\text{SS}}$  forces 8741A to single step. It is useful in debugging. For every instruction executed 8741A sends out a pulse on SYNC pin. Other pins have their usual meaning.

### 3.2.2 Interfacing the Multiplier Accumulator, TDC 1008:

TDC 1008 is a high speed TTL based LSI device capable of 8x8 bit multiplication and product accumulation in 70 to 115 nsecs. The numerical system can be 2's complement or unsigned magnitude. The output contents can be added to or subtracted from the next product. The accumulate function can be disabled for multiply only. The output registers can be initialised (preloaded) before multiply operation. These features make TDC 1008 useful in implementing the programmable digital filters for our noise sources. The logic diagram of TDC 1008 is shown in Fig. 3.2.3.

The pin out of TDC 1008 is also given in Appendix 'D'.  $X_{in}$  and  $Y_{in}$  are the two 8 bit inputs. XTP, MSP and LSP are the 3 bit extended product, 8 bit most significant product and 8 bit least significant product respectively. The output can be read as a 19 bit word or as three separate 8 bit.

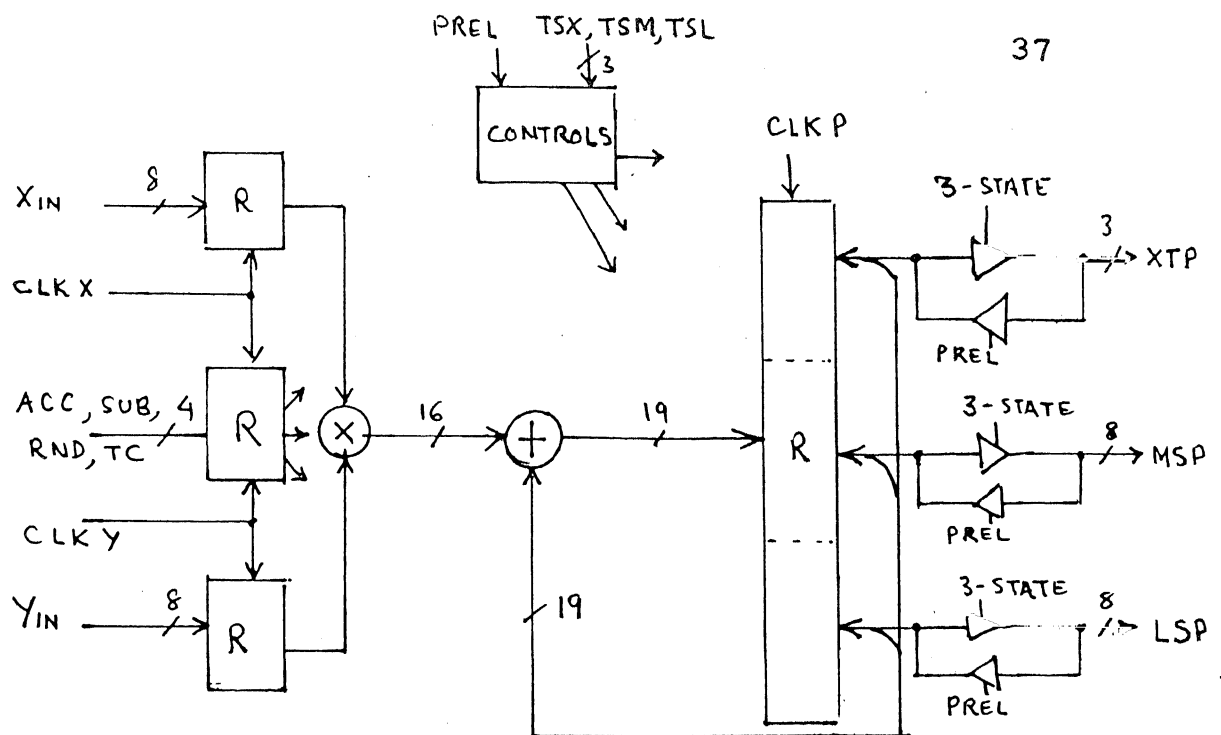


Fig. 3.2.3: Logic Diagram of TDC 1008

words due to the limitation of 8 bit microcomputers. There are three clock inputs Clk X, Clk Y, and Clk P for loading the 8 bit inputs and reading the product. The control signals two's compliment (TC), round control (RND), accumulate function (ACC) and subtract function (SUB) are latched through the 8212 simple output port shown in Figure 3.2.1. Other control signals - PREL, TSX, TSM and TSL are derived from the decoder logic as per the truth table shown as Table 5.

In Table 5, Q stands for output buffers at low impedance. The contents of output registers will be transferred to output pins. P stands for preload data. Output buffers at high impedance, or output disabled and data supplied externally at output pins will be loaded into the output registers at the

Table 5: Preload Truth Table

PREL	TSX	TSM	TSL	XTP	MSP	LSP
0	0	0	0	Q	Q	Q
0	0	0	1	Q	Q	H
0	0	1	0	Q	H	Q
0	0	1	1	Q	H	H
0	1	0	0	H	Q	Q
0	1	0	1	H	Q	H
0	1	1	0	H	H	Q
0	1	1	1	H	H	H
1	0	0	0	H	H	H
1	0	0	1	H	H	P
1	0	1	0	H	P	H
1	0	1	1	H	P	P
1	1	0	0	P	H	H
1	1	0	1	P	H	P
1	1	1	0	P	P	H
1	1	1	1	P	P	P

rising, edge of Clk P. H represents high impedance state. In the present work SUB and RND are kept low (0) and TC and ACC are made high (1) when 2's complement multiplication and accumulation are both needed. This is done by sending suitable words to the 8212 simple output latch. Words are formed by the 8 bits of the data word shown in Figure 3.2.4.

EN CLKP	X	X	$\overline{SS}$	TC	RND	ACC	SUB
------------	---	---	-----------------	----	-----	-----	-----

Fig. 3.2.4: Data word for 8212

### 3.2.3 Decoder Logic for White Gaussian Noise Generator Card:

The decoder logic for the white Gaussian noise generator card is derived from the truth tables of Tables 4 and 5. The block diagram is shown in Figure 3.2.5. A 3-8 decoder (74 LS138) is used to generate chip selects ( $\overline{CS}$ ) for 8741A, 8212 and seven control signals Clk X, Clk Y, Clk P, PREL, TSX, TSM and TSL for the TDC 1008 with the help of additional AND, OR, EX-OR and NAND gates. The chip select ( $\overline{CS}$ ) for EPROM is also shown in Figure 3.2.5. The address map of the white Gaussian noise card is shown in Table 6. The circuit diagram for this card is shown in Appendix 'E'.



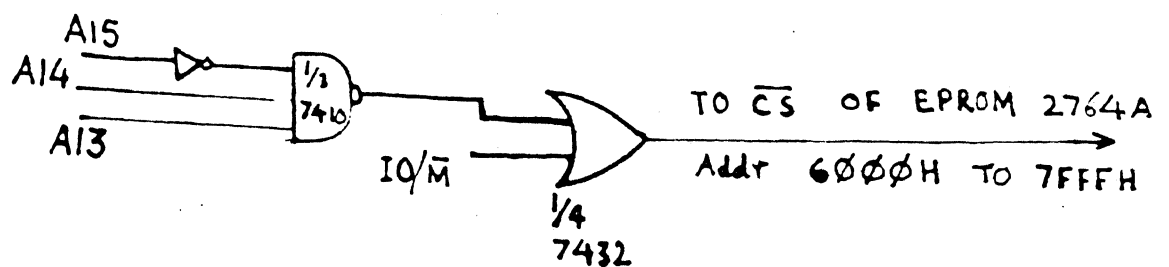
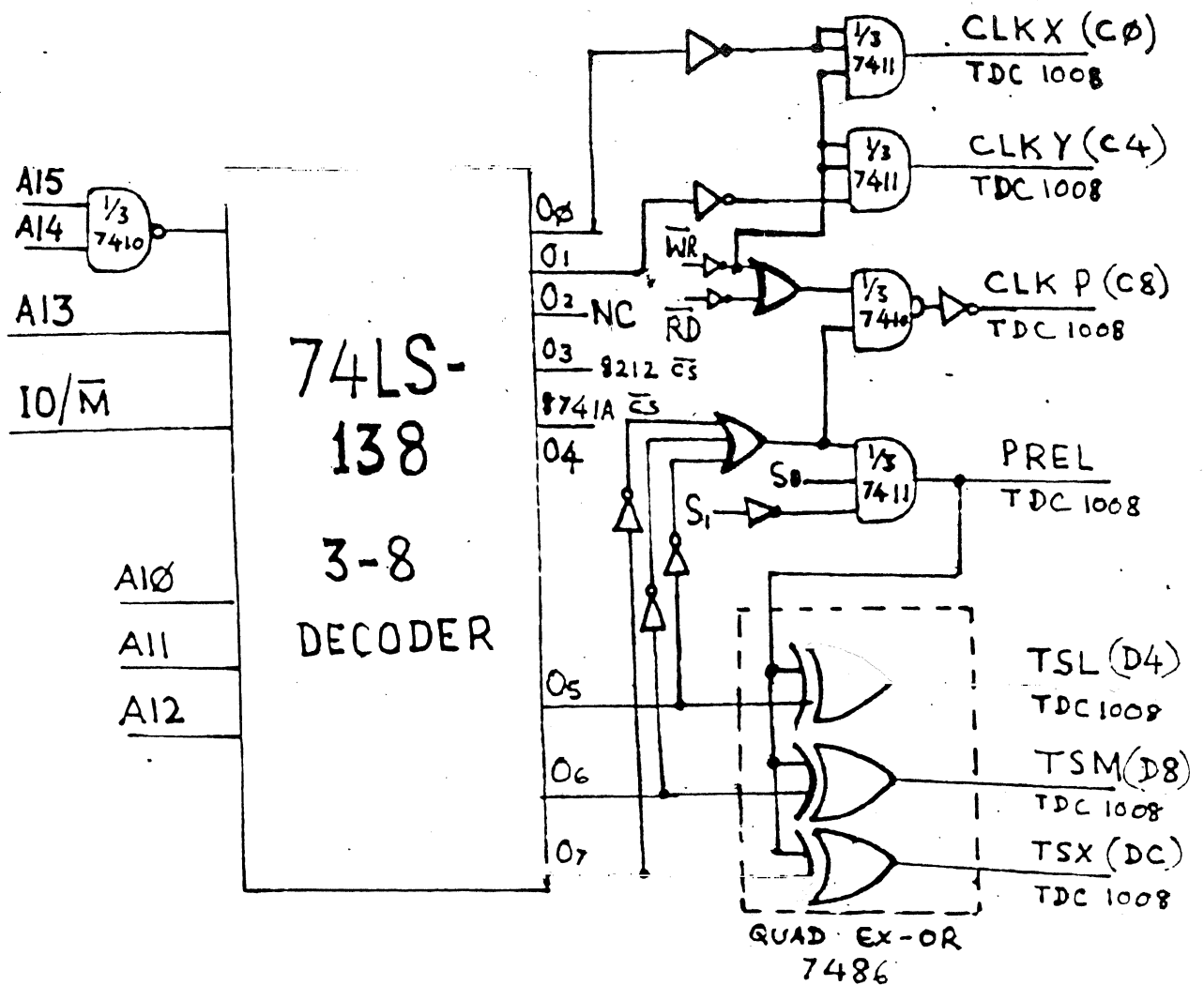


Fig. 3.2.5: Decoder logic for the white Gaussian noise generator card.

Table 6: Address Map of the Noise Generator Card.

Sl. No.	Address	Device/Function
1.	6000H to 7FFFH	EPROM (2764A)
2.	C0H to C3H	Clk X for TDC 1008 (X-INPUT WRITE ONLY)
3.	C4H to C7H	Clk Y FOR TDC 1008 (Y-INPUT WRITE ONLY)
4.	C8H to CBH	SPARE
5.	CCH to CFH	$\overline{CS}$ FOR 8212 (WRITE ONLY)
6.	DDH to D3H	$\overline{CS}$ FOR 8741A (READ ONLY)
7.	D4H to D7H	LTP FOR TDC 1008 (READ/WRITE)
8.	D8H to DBH	MTP FOR TDC 1008 (READ/WRITE)
9.	DCH to DFH	XTP FOR TDC 1008 (READ/WRITE)

### 3.3 THE OUTPUT CARD:

The digital noise samples are distributed as various

noise sources  $h_{kc}(t)$  and  $h_{ks}(t)$  depending on how many taps are required by the user. For the present work we have chosen five taps per diversity channel. With these parameters in mind a layout of the output card is shown in Figure 3.3.1.

The output card caters for ten processes (two per tap). Therefore ten DACs are shown followed by 10 low pass filters to smoothen the analog output. To hold the digital samples constant at the input of the DACs, ten simple output ports derived from three programmable 8255 I/O ports and one 8212 port are included in the output card. The decoder logic for this card is shown in Figure 3.3.2.

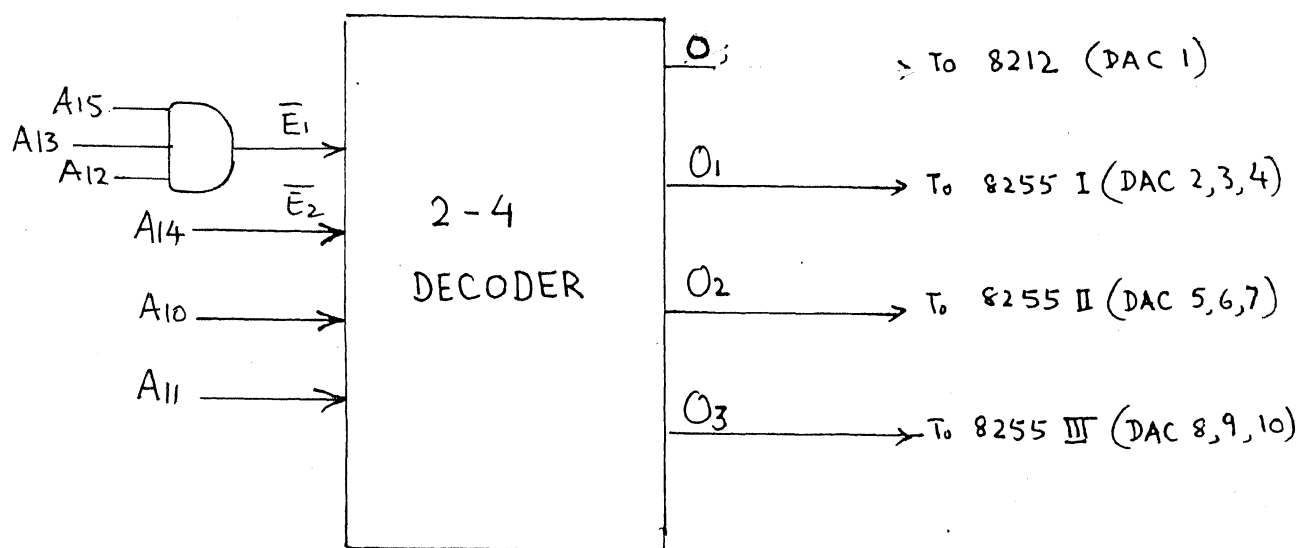


Fig. 3.3.2: Decoder Logic for the Output Card.

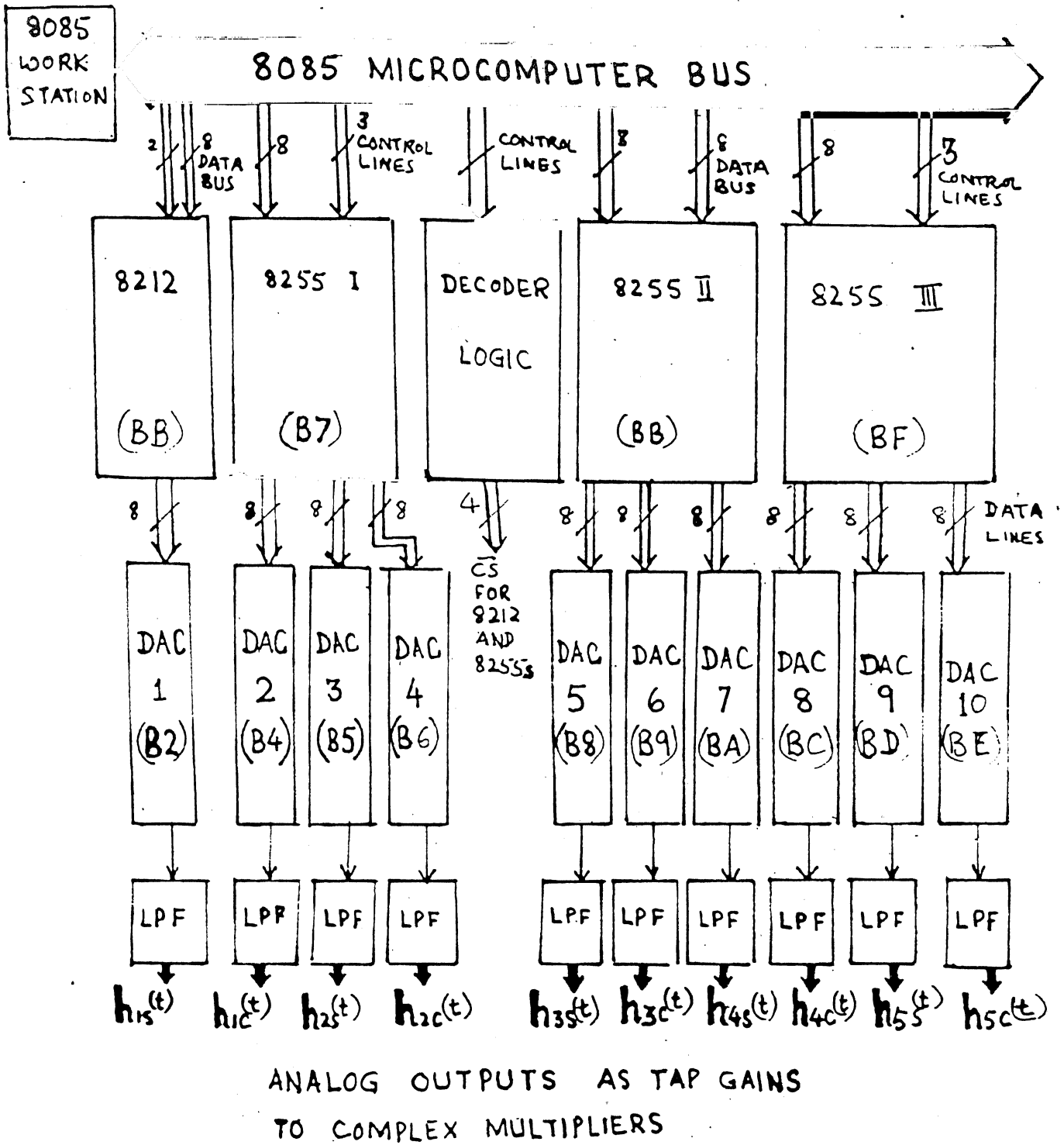


Fig. 3.3.1: Layout of the output card

A 2-4 decoder produces the four chip select ( $\overline{CS}$ ) Signals for the four I/O devices. The address map of the output card is shown in Table 7.

Table 7: Address Map of Output Card.

Sl. No.	Address	DAC No.	I/O Device
1.	B2H	1	8212 Data
2.	B3H	-	8212 Command word
3.	B4H	2	8255 (I) Data
4.	B5H	3	8255 (I) Data
5.	B6H	4	8255 (I) Data
6.	B7H	-	8255 (I) Command word
7.	B8H	5	8255 (II) Data
8.	B9H	6	8255 (II) Data
9.	BAH	7	8255 (II) Data
10.	BBH	-	8255 (II) Command word
11.	BCH	8	8255 (III) Data
12.	BDH	9	8255 (III) Data
13.	BEH	10	8255 (III) Data
14.	BFH	-	8255 (III) Command word

## CHAPTER 4

### SOFTWARE IMPLEMENTATION

One can have a rough guess, from Figure 2.2.2, of the hardware involved in the construction of eight parallel PRBSGs. This job, on the other hand, is done by a single microcomputer chip 8741A using its memory locations as shift registers. The software implementation of these PRBSGs and the generation of noise sources is the topic of discussion in this chapter.

#### 4.1 SOFTWARE FOR THE WHITE GAUSSIAN NOISE GENERATOR:

An algorithm evolved from Sections 2.2.3 and 2.3 is shown in the form of a flow chart in Figure 4.1.1. In this section we will show how eight bit addresses are generated with uniform probability by the use of eight parallel PRBSGs, and how random data (of Table 2) is accessed from the look up table formed in the ROM of 8741A.

##### 4.1.1 Implementation of the PRBSGs:

The 8741A single chip microcomputer has 64 RAM locations (address 00H to 3FH) in addition to 1 K bytes of ROM (address 000H to 3FFH). The organisation of the internal RAM is shown in Figure 4.1.2.

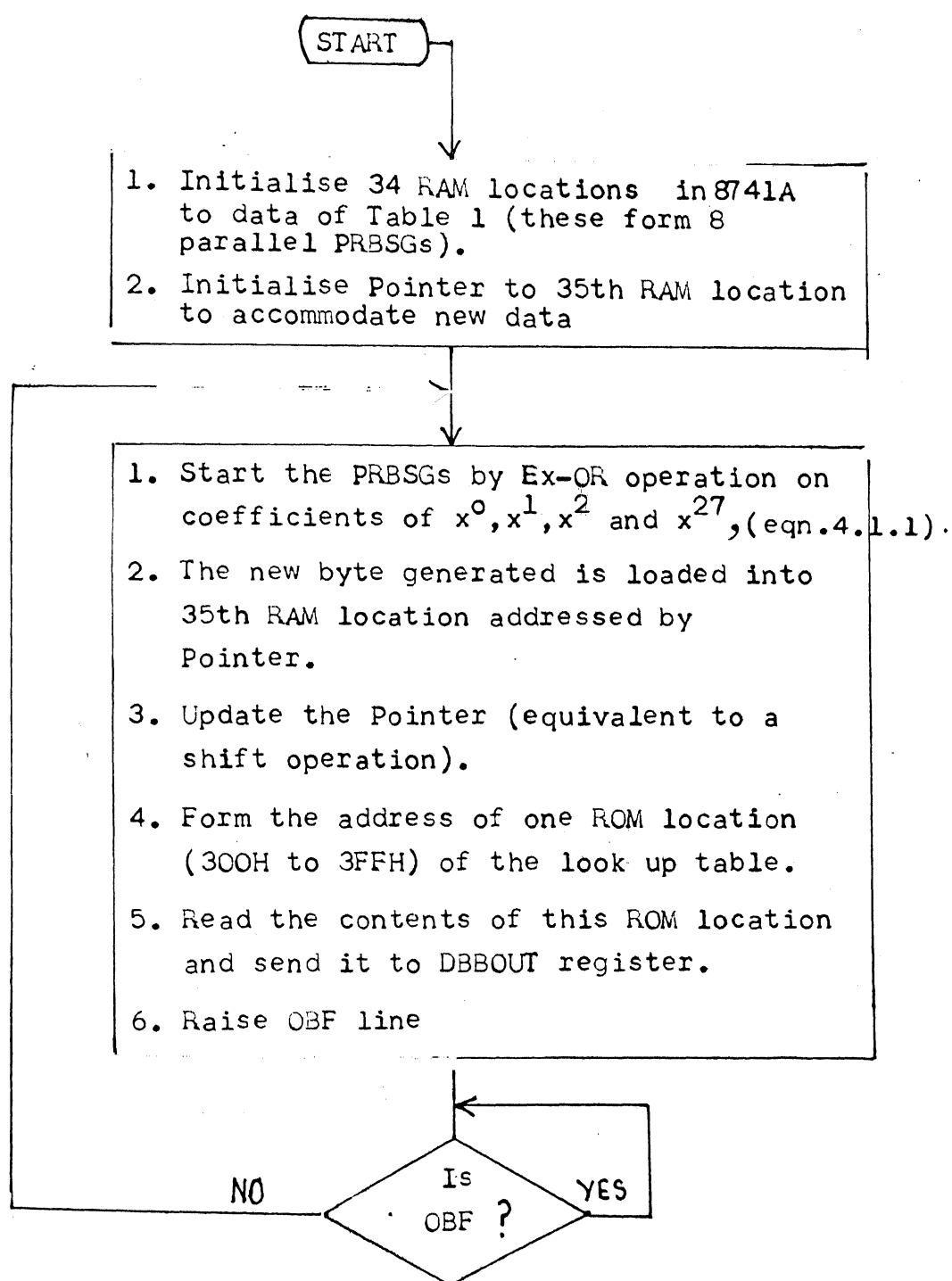


Fig. 4.1.1: Algorithm for white Gaussian Noise Generation.

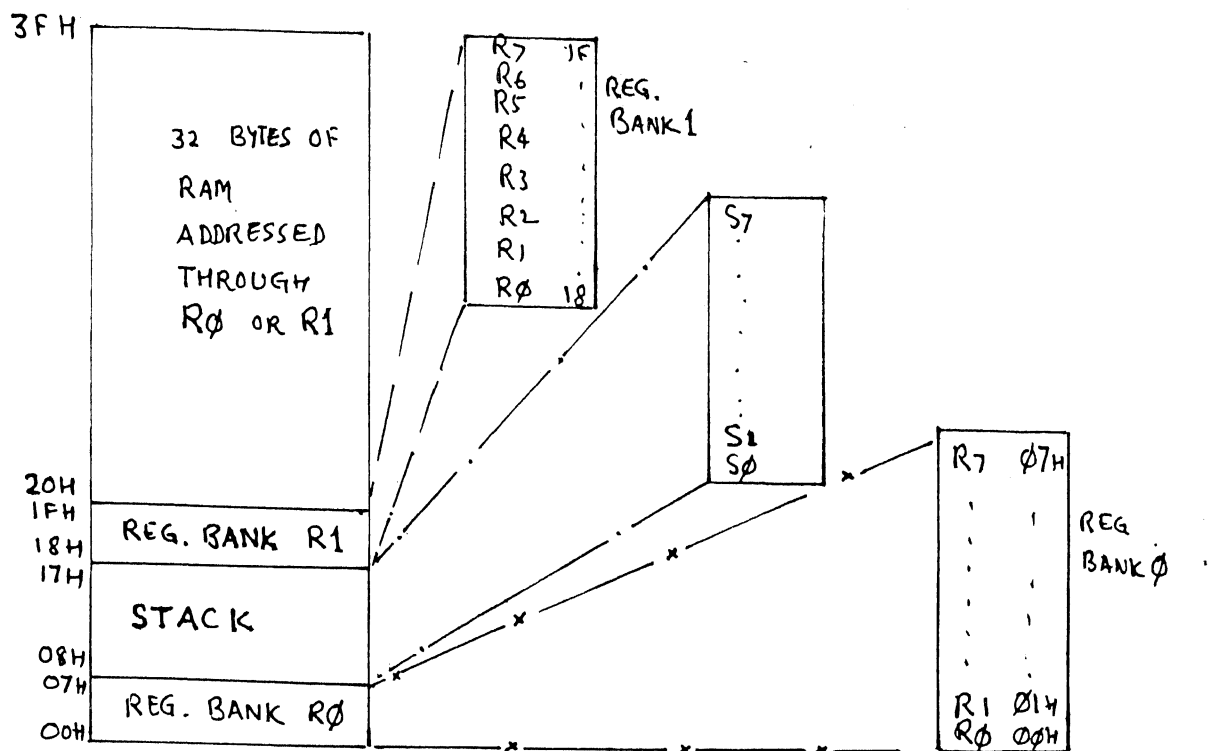


Figure 4.1.2: Organisation of Internal RAM (8741A)

Equation 2.2.2 selected for implementing the PRBSG is reproduced here for easy reference

$$p(x) = x^{34} \oplus x^{27} \oplus x^2 \oplus 1 \quad (4.1.1)$$

The feed back logic of (4.1.1) is shown in Figure 4.1.3.

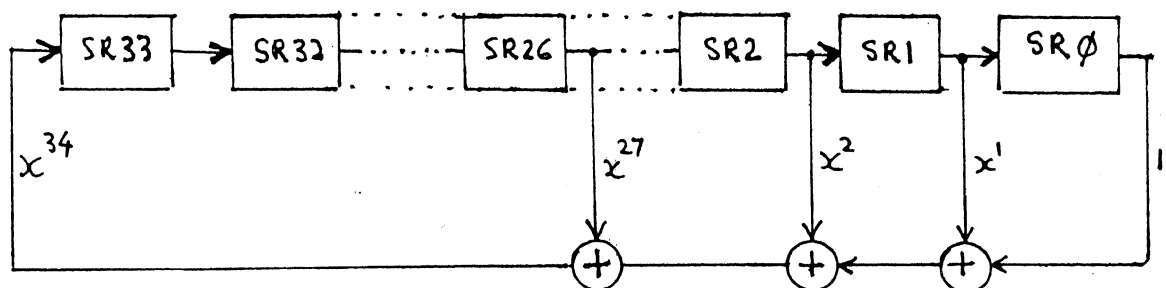


Figure 4.1.3: Feedback logic of PRBSG



Modulo 2 operations are carried out on two bits at a time to generate the feed back bit at each right shift of the 34 shift registers. At each right shift, the least significant bit (content of SRO) is lost while others are shifted to the right to accomodate the new feed back bit, which is also the output of the PRBSG. Eight such PRBSGs in parallel will give one byte which will form the lower eight bits of the address of the memory location in ROM area. The most significant nibble is fixed as 3 as we have chosen to form the look up table from 300H to 3FFH. Thus, we are able to address the 256 memory locations in the look up table. The output of the eight PRBSGs is an integer from [0,255] having equal probability of occurrence as discussed in Section 2.2.3. In Figure 4.1.4 we show how the eight PRBSGs each representing (4.1.1) is implemented.

The contents of the 34 shift registers (address 1EH to 3FH) shown in Fig. 4.1.4 are initialised to data of Table 1 to ensure statistical independence as discussed in Section 2.3. Also, attention is drawn to Figure 4.1.1 in this connection. A pointer is maintained in this algorithm to point to the RAM location where the new byte generated after modulo 2 (or equivalently EX-OR) operation is to be stored to represent the contents of the 34th shift register for the next operation. Thus initially the pointer points to 1DH which is as shown

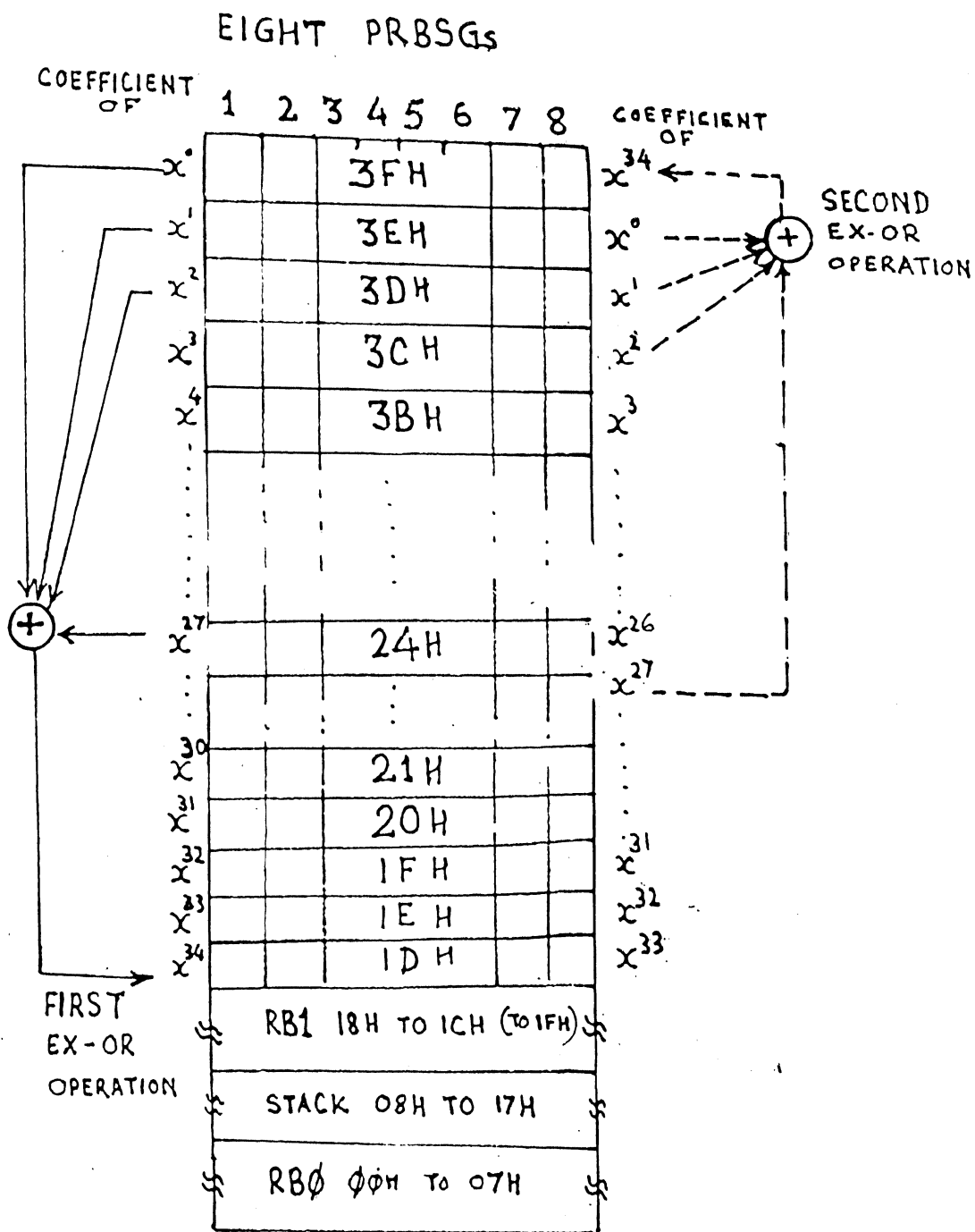


Fig. 4.1.4 Eight PRBSGs in RAM Area of 8741A.

in Figure 4.1.4. The result thus goes into this RAM location first time. Now the pointer is manipulated to point to 3FH as shown in dotted line in Figure 4.1.4. In right shift operation the coefficient of  $x^0$  is lost as pointed out earlier in this section. Hence this RAM location is free to accommodate the next data generated as a result of EX-OR operation on the coefficients of  $x^0$ ,  $x^1$ ,  $x^2$  and  $x^{27}$ . In this way the pointer is manipulated and the eight parallel PRBSGs are implemented using 35 RAM location (1DH to 3FH) only. The coefficients of  $x^0$ ,  $x^1$ ,  $x^2$  and  $x^{27}$  are addressed simply by adding suitable numbers to the address held by the pointer. However, it is to be ensured that the calculated address always points to RAM location loop formed by RAM addresses 1DH to 3FH. In effect the pointer rolls over in this loop generating at each operation, 8 bit words from [0, 255].

#### 4.1.2 Accessing the Look up Table:

The data having Normal distribution as calculated in Section 2.4 is stored permanently in ROM locations 300H to 3FFH. The 8 bit data generated by EX-OR operation explained above forms the lower byte 00H to 1FH. The upper nibble is kept fixed as 3, thus addressing the ROM area 300H to 3FFH. The contents are put out to the DBBOUT register shown in Figure 3.2.2. 8741A now waits for the external processor 8085 to read this data under interrupt control

98553

provided by OBF line as explained earlier in Section 3.2.1. The 8741A checks for OBF line to go low by the instruction JOBF. When the output buffer becomes empty as a result of a read operation by 8085, 8741A proceeds to work out the next sample as explained above. Software listing of the 8741A Assembly language program is attached as Appendix 'F'. The program execution starts from 000H when 8741A is reset (by pressing RESET switch on 8085 workstation).

#### 4.2 SOFTWARE FOR THE FIRST ORDER DIGITAL FILTERS:

Partly the digital filtering is carried out by the multiplier accumulator TDC 1008 discussed in Section 3.2.2. Since ten digital filters are to be implemented, one corresponding to each process  $h_{kc}(t)$  or  $h_{ks}(t)$ , multiplexing on TDC 1008 is necessary. 8085 workstation RAM area is used to store current samples for feed back once the next samples are transmitted to a particular DAC. Ten RAM locations (address 5721H to 572AH) hold these samples. For each digital filter different feedback gain  $D$ , as explained in Section 2.4, is used to obtain different bandwidth on each tap. The values of  $D$  for each process are read from Table 3 (address 6FOOH to 6FFFH) and stored in RAM locations (5711H to 571AH) for implementing the digital filter shown in Figure 2.4.2. Hence, digital samples having Normal distribution are read one by one by 8085, processed through digital filters using

TDC 1008 and distributed to DACs (Figure 3.3.1 ) whose output is then smoothened by analog filters to produce the desired  $h_{kc}(t)$  or  $h_{ks}(t)$ . A delay is imposed between samples to ensure uniform sampling rate. This algorithm is shown in the form of a flow chart in Figure 4.2.1. For interaction between the user and the simulator certain messages are printed on the screen. The user is then required to feed the desired parameters such as the number of taps, sampling period (in milliseconds) and the bandwidth (in Hzs) on each tap. The user may change the parameters, when-ever he desires, by pressing any key (except the ESC key). On pressing the ESC key the simulation will be terminated. A program in 8085 Assembly language to implement the algorithm discussed above is given in Appendix 'G'. The origin of the program is 65CEH. To start the program GO65CE command has to be given to the 8085 workstation. The simulator then asks the parameters required by the user. Once the parameters are given, the simulator goes into action. It can only be stopped by pressing any of the keys as explained earlier. Since interrupt RST 6.5 is used to read the 8741A a JMP 653A must be written at RAM location ~~58~~12 before giving the GO command. The output of 8741A (before filtering) can be watched on DAC-7 if a JMP 675F instruction is written at the RAM location ~~S~~812 instead of JMP 653A.

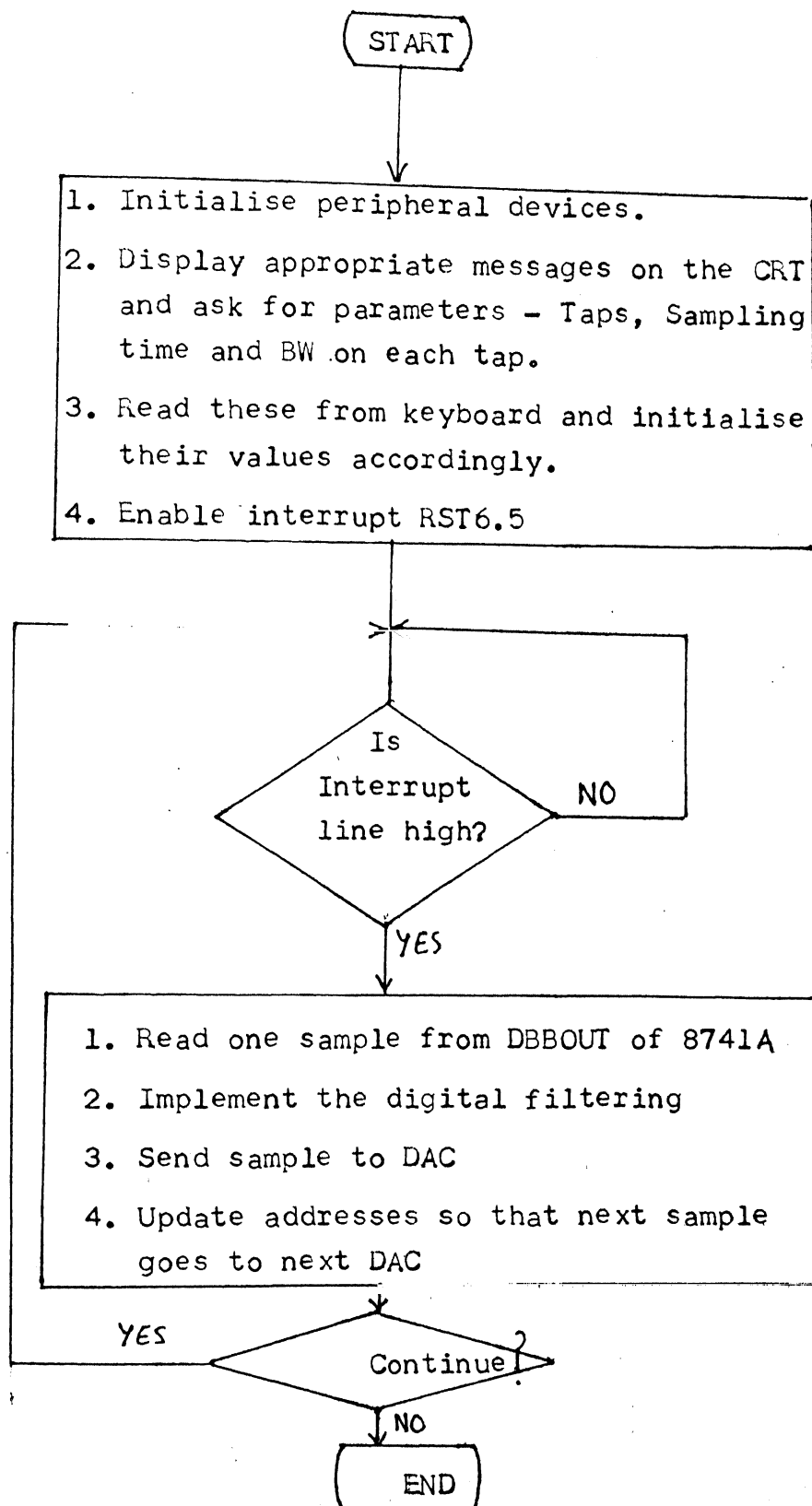


Fig. 4.2.1: Algorithm for the Digital Filtering.

## CHAPTER 5

### CONCLUSION

The single chip microcomputer implementation of the PRBS based Gaussian noise generator is capable of generating 4000 independent noise samples every second. Typically, a full fledged quadruple diversity, ten tap Tropo-channel simulator requires 80 ( $4 \times 10 \times 2$ ) independent noise sources with each having a bandwidth of upto 50 Hz for simulating the fastest possible fades expected in the channel. A 50 Hz noise bandwidth can be practically realised by feeding 200 samples of independent Gaussian noise per second through appropriate digital filters. Thus for a full ten tap, quadruple diversity system a noise sample rate of 16,000/sec is required. Accordingly for a five tap system 8000 samples/sec would be adequate.

The present implementation, with 4000 independent Gaussian noise samples per second, can cater to a five tap, dual diversity simulator or a quadruple diversity simulator with fade rate restricted to 25 Hz. An evaluation of the time taken for digital filtering using TRW, TDC-1008 multiplier-accumulator (MAC) shows that it is well within

the 250  $\mu$ sec noise sample period achieved by the 8741A. As an experiment a five tap, single diversity channel simulator was implemented and ten processes  $h_{kc}(t)$  and  $h_{ks}(t)$  were generated. The output card thus has 10 DACs to cater to these 10 processes only. If larger number of noise sources are required, then the output card must be suitably modified. In fact the number of DACs is equal to the number of processes generated. In view of this, signal processing ICs such as I2920 could be investigated to reduce the hardware.

In comparison to the noise generators of the Tropo-simulator [8] fabricated earlier in this Institute's Advanced Centre for Electronic Systems, the present implementation scores over it in smaller chip count, lowpower consumption, higher reliability and better user convenience.

With eight bit restriction on the word length of 8741A and 8085 we achieved peak Gaussian noise samples of  $2.65\sigma$  as explained in Section 2.3. For better noise quality, single chip microcomputers having 16 bit word length can be used to produce typical peak sample values of  $4\sigma$ . However, these noise samples must have a resolution of 12 bits and therefore requires 12 bit MAC (such as TDC 1009) for implementing the digital filtering. The multiplier Accumulator will have to be driven by a 16 bit microcomputer such as 6800.



As a suggestion for future implementation, the present work can be modified by making 8085 processor a part of the noise generator card of Figure 3.2.1. As a stand alone card, it can interact with a personal computer (PC) through a suitable serial communications I/O port. The PC can be used to program the noise generators and also for monitoring and computational purpose.

## REFERENCES

1. Andreas Antonion, 'Digital Filters Analysis and Design', McGraw Hill Book Company, New York, 1979, p. 14.
2. George S. Fishman, 'Concepts and Methods in Discrete Event Digital Simulation', John Wiley and Sons, New York, 1973, pp. 167-171.
3. Ibid., p. 184.
4. Ibid., p. 211.
5. Murray R. Spiegel, 'Mathematical Handbook of Formulas and Tables', McGraw Hill, New York, 1968, Table 47, p. 257.
6. Peterson and Weldon, 'Error Correcting Codes', M.I.T. Press, New York, 1978, pp. 182-188.
7. Ibid., Appendix 'C', p. 492.
8. P.R.K. Rao, 'Fading Dispersive Channel Simulator', a project proposal (July 1976) and two follow up reports (1978 and 1979) submitted to Ministry of Defence/ Govt. of India by I.I.T. Kanpur.

Appx 'A'

LOC	OBJ	LINE	SOURCE STATEMENT
5000		1	ORG 5000H ; RUN 34 SR PRBSG AND
5000	210055	2	LXI H, 5500H ; TO RECORD IT'S STATE
5003	220057	3	SHLD 5700H ; EVERY (8000H * FFFFH)
5006	11FFFF	4	LP1: LXI D, 0FFFFH ; COUNT=FFFFH
5009	010080	5	LP2: LXI B, 8000H ; COUNT=8000H * FFFFH
500C	2A0057	6	PRBS: LHLD 5700H ; POINT TO 34th SHIFT REGISTER (SR)
500F	3E06	7	MVI A, 06H
5011	85	8	ADD L
5012	6F	9	MOV L, A ; POINT TO 27th SR
5013	46	10	MOV B, M
5014	3E19	11	MVI A, 19H
5016	85	12	ADD L
5017	6F	13	MOV L, A ; POINT TO 2nd SR
5018	78	14	MOV A, B
5019	AE	15	XRA M ; MODULO 2 ON 27th AND 2nd SRs
501A	2C	16	INR L
501B	AE	17	XRA M ; ---- DO --- (A) AND 1st SR
501C	2C	18	INR L
501D	AE	19	XRA M ; ---- DO --- (A) AND 0th SR
501E	2A0057	20	LHLD 5700H ; ADDR OF NEW SR
5021	0B	21	DCX B
5022	C20C50	22	JNZ PRBS ; REPEAT 8000H TIMES
5025	1B	23	DCX D ; REPEAT ABOVE LOOP FFFFH TIMES
5026	C20950	24	JNZ LP2
5029	013550	25	LXI B, MSG
502C	CD5E00	26	CALL 005EH ; PRINT MSG
502F	CD3700	27	CALL 0037H ; READ SUB: TO CONTINUE PRESS KEY
5032	C30650	28	JMP LP1 ; OTHERWISE PRESS CNTL C
		29	; IN MONITER MODE REGISTER CONTENTS
		30	; CAN BE READ FROM ADDR 5500 TO 55FF
		31	; SA IS IN MEM LOC 5700, 01
5035	52454144	32	MSG: DB 'READ SRs *'
5039	20535273		
503D	20202A		
		33	END

PUBLIC SYMBOLS

EXTERNAL SYMBOLS

USER SYMBOLS

LP1	A 5006	LP2	A 5009	MSG	A 5035	PRBS	A 500C
-----	--------	-----	--------	-----	--------	------	--------

ASSEMBLY COMPLETE, NO ERRORS

LOC	OBJ	LINE	SOURCE STATEMENT
5000		1	ORG 5000H ; PGME TO RUN 34 BIT PRBSG
5000	3E08	2	MVI A, 08H ; COUNT = 8
5002	320057	3	STA 5700H
5005	3E1B	4	MVI A, 1BH ; UNMASK RST 7.5 INTERRUPT
5007	30	5	SIM
5008	3E81	6	MVI A, 81H ; INITIALISE 8255
500A	D3BB	7	OUT 0BBH
500C	3E22	8	MVI A, 22H ; RESET SRs, ENABLE CLK, DISABLE
		9	INPUT TO SRs AND CTR 1
500E	D3BA	10	OUT 0BAH
5010	3E33	11	MVI A, 33H ; SRs = 1
5012	D3BA	12	OUT 0BAH
5014	3E74	13	MVI A, 74H ; INITIALISE CTR 1
5016	D3A3	14	OUT 0A3H
5018	3EFF	15	MVI A, 0FFH
501A	D3A1	16	OUT 0A1H
501C	D3A1	17	OUT 0A1H ; N1 = FFFFH
501E	3E30	18	MVI A, 30H ; CTR 0, MODE 0, BOTH BYTES, BINARY
5020	D3A3	19	OUT 0A3H
5022	3E00	20	WAIT1: MVI A, 00H ; N0=4000H
5024	D3A0	21	OUT 0A0H
5026	3E40	22	MVI A, 40H
5028	D3A0	23	OUT 0A0H
502A	3EFF	24	MVI A, 0FFH ; ENABLE CTR 1, INPUT TO SRs, CLK
502C	D3BA	25	OUT 0BAH ; START PRBSG
502E	FB	26	EI
502F	CD3700	27	WAIT2: CALL 0037H ; READ SUB: WAIT FOR INTR RST 7.5
5032	CD5E00	28	CALL 005EH ; PNTMS SUB
5035	3A0057	29	LDA 5700H
5038	3D	30	DCR A ; COUNT = (COUNT-1)
5039	320057	31	STA 5700H
503C	C22250	32	JNZ WAIT1 ; 8 READINGS OVER ?
503F	016550	33	OVER: LXI B, M2
5042	CD5E00	34	CALL 005EH ; PNTMS SUB
5045	CF	35	RST 1
5046	3E0D	36	ISR: MVI A, 0DDH
5048	D3BA	37	OUT 0BAH ; DISABLE CLK TO SRs
504A	015A50	38	LXI B, M1
504D	CD5E00	39	CALL 005EH ; PNTMS SUB
5050	CD4600	40	CALL 0046H ; CRLF SUB
5053	3E1B	41	MVI A, 1BH ; UNMASK RST 7.5
5055	30	42	SIM
5056	CD3700	43	CALL 0037H ; READ SUB: READ SRs CONTENTS NOW AND
		44	PRESS KEY
5059	C9	45	RET
505A	20524541	46	M1: DB / READ SRs* /
505E	44205352		
5062	732A20		
5065	20382052	47	M2: DB / 8 READINGS OVER* /
5069	45414449		
506D	4E475320		
5071	4F564552		
5075	2A20		
		48	END

LOC	OBJ	LINE	SOURCE STATEMENT
-----	-----	------	------------------

PUBLIC SYMBOLS

EXTERNAL SYMBOLS

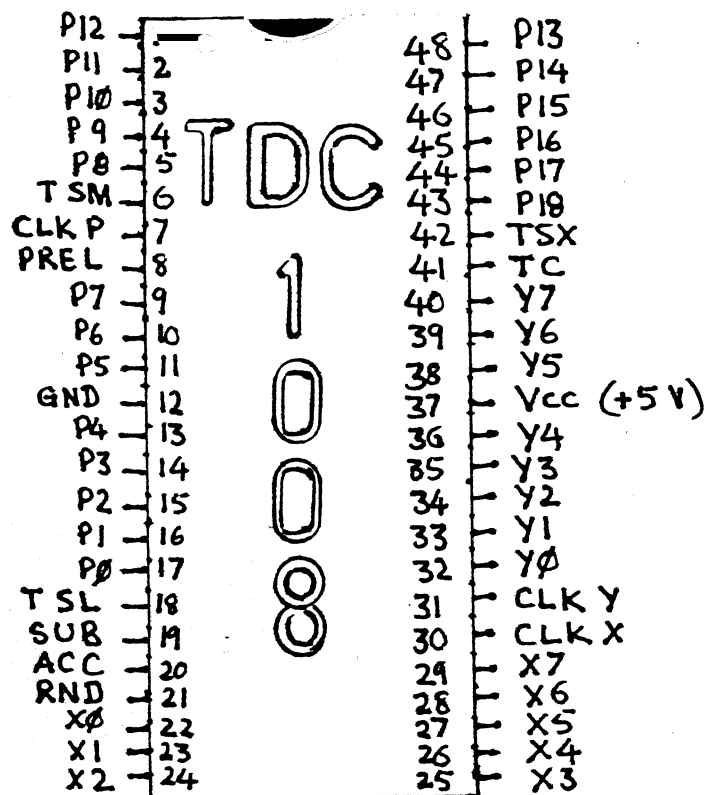
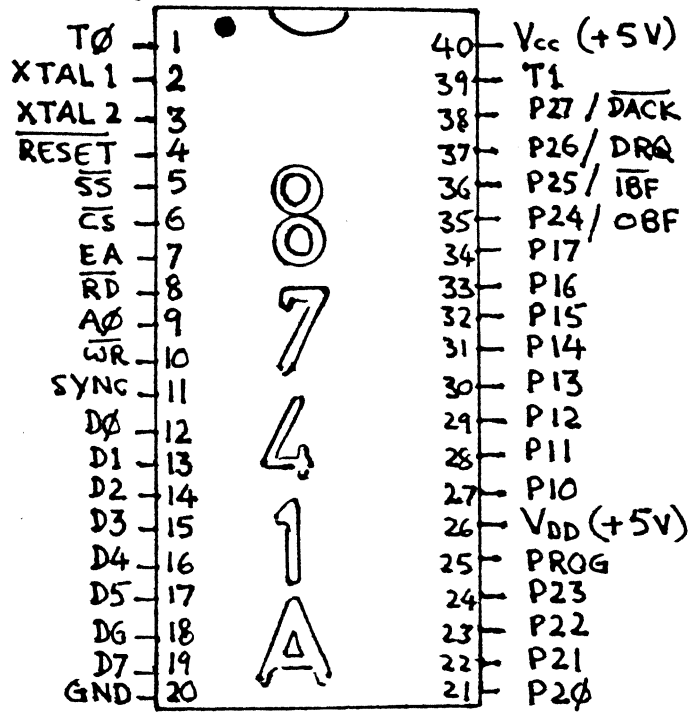
USER SYMBOLS

ISR	A 5046	M1	A 505A	M2	A 5065	OVER	A 503F	WAIT1	A 5022
-----	--------	----	--------	----	--------	------	--------	-------	--------

ASSEMBLY COMPLETE, NO ERRORS

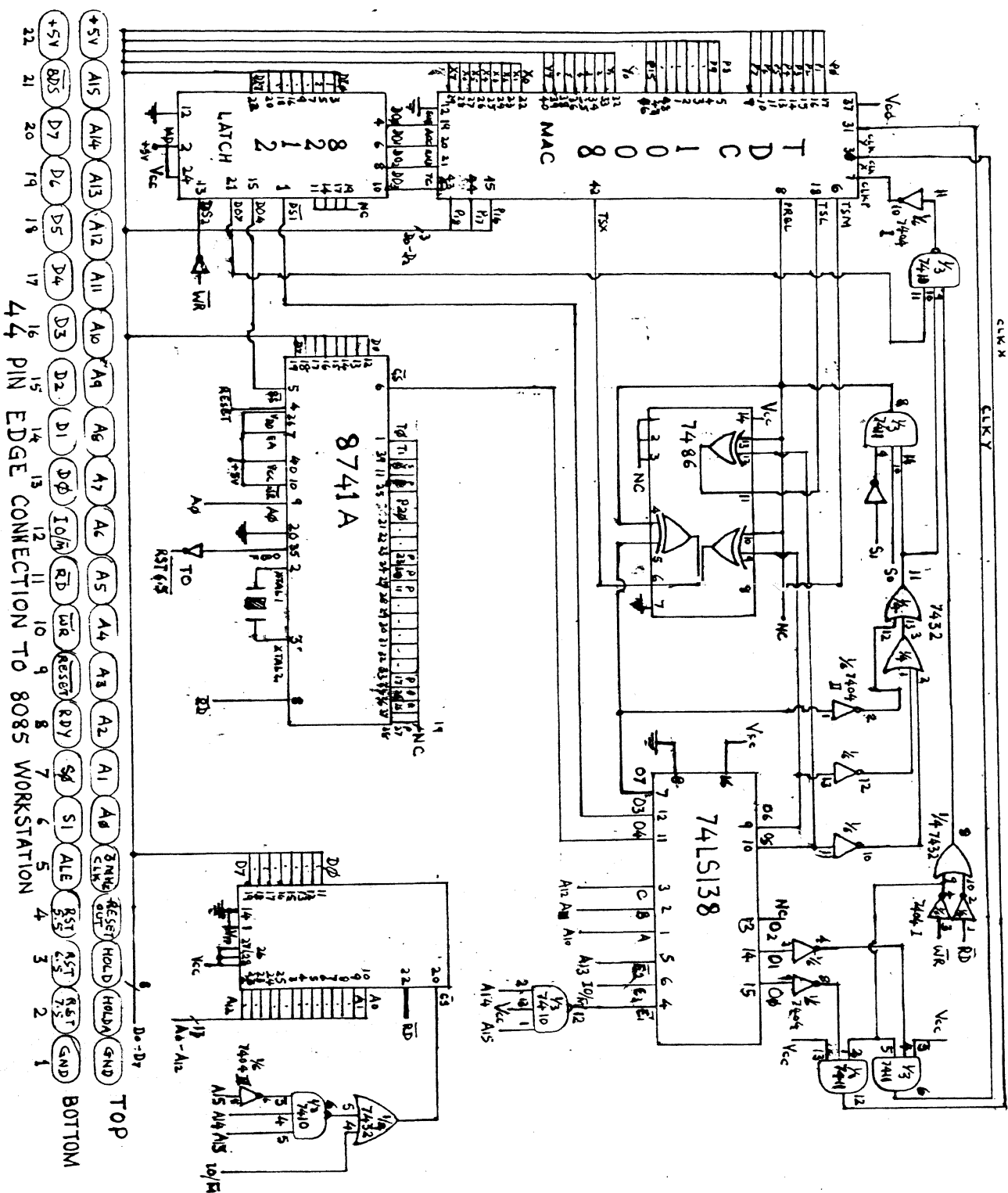
# PINOUT OF 8741A AND TDC 1008

ATTENDIA 11



# CIRCUIT DIAGRAM - NOISE GENERATOR CARD

APPENDIX-E



Appx 'F'

LDC OBJ	LINE	SOURCE STATEMENT
0000	1	ORG 000H ; GO ON ON RESET
0000 D5	2	SEL RB1 ; REG BANK 1
0001 1453	3	CALL INIT ; SUBROUTINE TO INITIALISE
0003 B91E	4	MOV R1, #1EH ; ADDR OF INPUT REG
0005 142C	5	NEXT: CALL SUBXOR ; SUBROUTINE TO DO EXOR OPS
0007 E3	6	MOV P3 A, @A ; FETCH SAMPLE FM ROM PAGE 3
0008 02	7	OUT DBB, A ; OUTPUT SAMPLE N(O, VARIANCE)
0009 23FF	8	MOV A, #0FFH ; WRITING 1 IN P24 DBF
000B 3A	9	OUTL P2, A ; OUT TO PORT 2
000C F5	10	EN FLAGS ; ENABLE INTERRUPT TO 8085
( 0)		
000D 00	11	WAIT: NOP ;
000E 860D	12	JOBF WAIT ; LOOP TO CHECK DBF FLAG
0010 C9	13	DEC R1 ; POINTER=POINTER-1
0011 1417	14	CALL SUB1D ; MANIPULATE POINTER IF REQUIRED
0013 FA	15	MOV A, R2 ; NEW DATA GEN IS FED TO INPUT REG
0014 A1	16	MOV @R1, A ;
0015 0405	17	JMP NEXT ; OUTPUT NEXT GAUSSIAN SAMPLE AND WAIT
0017 231D	18	SUB1D: MOV A, #1DH ; SUBROUTINE TO CHECK POINTER ADDR
0019 37	19	CPL A ; 1'S COMPLIMENT OF (A)
001A 17	20	INC A ; 2'S ---DO-----
001B 69	21	ADD A, R1 ; (R1)-1DH=(A)
001C 9620	22	JNZ NOCHG ; ADDR NOT=1DH
001E B93F	23	MOV R1, #3FH ; POINTER ADDR IS MODIFIED
0020 93	24	NOCHG: RETR ; RESTORE PSW AND RETURN
0021 37	25	SUB8: CPL A ; 1'S COMPLIMENT
0022 17	26	INC A ; 2'S ----DO-----
0023 033F	27	ADD A, #3FH ; (A)+3FH - (A)
0025 F62B	28	JC INRNG ; NO CHANGE REQUIRED
0027 37	29	CPL A ; MODIFY ADDR
0028 031E	30	ADD A, #1EH ; BY 1EH
002A A9	31	MOV R1, A ; POINTER MODIFIED
002B 93	32	INRNG: RETR
002C F9	33	SUBXOR: MOV A, R1 ; SADDR
002D C5	34	SEL RBO ; REG BANK 0
002E A8	35	MOV R0, A ; SA IN R0 RBO
002F 0306	36	ADD A, #06H ; ADDR OF THE 27TH POWER OF X
0031 A9	37	MOV R1, A ; IN REG R1 RBO
0032 1421	38	CALL SUB8 ; SUBROUTINE TO MANIPULATE ADDR
0034 F1	39	MOV A, @R1 ; COEFICIENT IN R7
0035 AF	40	MOV R7, A
0036 F8	41	MOV A, R0
0037 031F	42	ADD A, #1FH ; ADDR OF 2ND POWER OF X
0039 A9	43	MOV R1, A ; IN REG R1 RBO
003A 1421	44	CALL SUB8 ; MODIFY R1 IF REQD
003C F1	45	MOV A, @R1
003D AE	46	MOV R6, A ; COEF IN R6
003E F8	47	MOV A, R0
003F 0320	48	ADD A, #20H ; SA+20H = ADDR OF X
0041 A9	49	MOV R1, A
0042 1421	50	CALL SUB8 ; MODIFY ADDR IF REQD
0044 F1	51	MOV A, @R1
0045 AD	52	MOV R5, A ; COEF OF X IN R5
0046 2321	53	MOV A, #21H ; SA + 21H = ADDR OF 0TH POWER OF X



LOC	OBJ	LINE	SOURCE STATEMENT
0048	68	54	ADD A, R0
0049	A9	55	MOV R1, A
004A	1421	56	CALL SUB8 ; MODIFY ADDR IF REQD
004C	F1	57	MOV A, @R1 ; COEF IN ACC
004D	DD	58	XRL A, R5
004E	DE	59	XRL A, R6 ; (A) + R5 + R6 + R7 = INPUT TO PRBS GE
004F	DF	60	XRL A, R7
0050	D5	61	SEL RB1
0051	AA	62	MOV R2, A ; BYTE GEN BY EXOR OP IN R2 RB1
0052	93	63	RETR
0053	B81E	64	INIT: MOV R0, #1EH ; TO ENTER DATA INTO MEM LOC 1EH TO 3FH
0055	B080	65	MOV @R0, #80H
0057	18	66	INC R0
0058	B0D0	67	MOV @R0, #0D0H
005A	18	68	INC R0
005B	B0C4	69	MOV @R0, #0C4H
005D	18	70	INC R0
005E	B0A5	71	MOV @R0, #0A5H
0060	18	72	INC R0
0061	B0CD	73	MOV @R0, #0CDH
0063	18	74	INC R0
0064	B088	75	MOV @R0, #88H
0066	18	76	INC R0
0067	B0B3	77	MOV @R0, #0B3H
0069	18	78	INC R0
006A	B096	79	MOV @R0, #96H
006C	18	80	INC R0
006D	B086	81	MOV @R0, #86H
006F	18	82	INC R0
0070	B09A	83	MOV @R0, #9AH
0072	18	84	INC R0
0073	B0C2	85	MOV @R0, #0C2H
0075	18	86	INC R0
0076	B0A2	87	MOV @R0, #0A2H
0078	18	88	INC R0
0079	B0EE	89	MOV @R0, #0EEH
007B	18	90	INC R0
007C	B084	91	MOV @R0, #84H
007E	18	92	INC R0
007F	B0B1	93	MOV @R0, #0B1H
0081	18	94	INC R0
0082	B0C1	95	MOV @R0, #0C1H
0084	18	96	INC R0
0085	B08F	97	MOV @R0, #8FH
0087	18	98	INC R0
0088	B086	99	MOV @R0, #86H
008A	18	100	INC R0
008B	B088	101	MOV @R0, #88H
008D	18	102	INC R0
008E	B080	103	MOV @R0, #80H
0090	18	104	INC R0
0091	B0C0	105	MOV @R0, #0C0H
0093	18	106	INC R0
0094	B084	107	MOV @R0, #84H
0096	18	108	INC R0

LOC	OBJ	LINE	SOURCE STATEMENT
0097	B08C	109	MOV @R0, #8CH
0099	18	110	INC R0
009A	B0CC	111	MOV @R0, #0CCH
009C	18	112	INC R0
009D	B088	113	MOV @R0, #88H
009F	18	114	INC R0
00A0	B090	115	MOV @R0, #90H
00A2	18	116	INC R0
00A3	B082	117	MOV @R0, #82H
00A5	18	118	INC R0
00A6	B08A	119	MOV @R0, #8AH
00A8	18	120	INC R0
00A9	B094	121	MOV @R0, #94H
00AB	18	122	INC R0
00AC	B088	123	MOV @R0, #88H
00AE	18	124	INC R0
00AF	B0A6	125	MOV @R0, #0A6H
00B1	18	126	INC R0
00B2	B08C	127	MOV @R0, #8CH
00B4	18	128	INC R0
00B5	B0C2	129	MOV @R0, #0C2H
00B7	18	130	INC R0
00B8	B0A0	131	MOV @R0, #0A0H ; ALL SHIFT REGS INITIALISED
00BA	93	132	RETR
0300		133	ORG 300H ; NEW ORIGIN
0300	7F55	134	DW 7F55H, 4D48H, 4542H, 403DH, 3C3AH, 3837H, 3634H, 3332H
0302	4D48		
0304	4542		
0306	403D		
0308	3C3A		
030A	3837		
030C	3634		
030E	3332		
0310	3130	135	DW 3130H, 2F2EH, 2D2CH, 2C2BH, 2A29H, 2928H, 2727H, 2625H
0312	2F2E		
0314	2D2C		
0316	2C2B		
0318	2A29		
031A	2928		
031C	2727		
031E	2625		
0320	2524	136	DW 2524H, 2323H, 2222H, 2120H, 2020H, 1F1FH, 1E1EH, 1D1DH
0322	2323		
0324	2222		
0326	2120		
0328	2020		
032A	1F1F		
032C	1E1E		
032E	1D1D		
0330	1C1C	137	DW 1C1CH, 1B1BH, 1B1AH, 1A19H, 1918H, 1817H, 1717H, 1616H
0332	1B1B		
0334	1B1A		
0336	1A19		
0338	1918		
033A	1817		

LOC	OBJ	LINE	SOURCE STATEMENT
033C	1717		
033E	1616		
0340	1615	138	DW 1615H, 1514H, 1414H, 1313H, 1212H, 1211H, 1111H, 1010H
0342	1514		
0344	1414		
0346	1313		
0348	1212		
034A	1211		
034C	1111		
034E	1010		
0350	100F	139	DW 100FH, 0F0FH, 0E0EH, 0D0DH, 0C0CH, 0B0BH, 0A0AH
0352	0F0F		
0354	0E0E		
0356	0D0D		
0358	0C0C		
035A	0B0B		
035C	0A0A		
035E	0909		
0360	0A0A	140	DW 0A0AH, 0909H, 0908H, 0808H, 0707H, 0706H, 0606H, 0605H
0362	0909		
0364	0908		
0366	0808		
0368	0707		
036A	0706		
036C	0606		
036E	0605		
0370	0505	141	DW 0505H, 0404H, 0403H, 0303H, 0202H, 0201H, 0101H, 0100H
0372	0404		
0374	0403		
0376	0303		
0378	0202		
037A	0201		
037C	0101		
037E	0100		
0380	0000	142	DW 0000H, 0FFFFH, 0FFFEH, 0FEFEH, 0FDFDH, 0FDFCH, 0FCFCH, 0FBFBH
0382	FFFF		
0384	FFFE		
0386	FEFE		
0388	FDFD		
038A	FDFC		
038C	FCFC		
038E	FBFB		
0390	FBFA	143	DW 0FBFAH, 0FAFAH, 0FAF9H, 0F9F9H, 0F8F8H, 0F7F7H, 0F7F6H, 0F6F6H
0392	FAFA		
0394	FAF9		
0396	F9F9		
0398	F8F8		
039A	7F77		
039C	F7F6		
039E	F6F6		
03A0	F5F5	144	DW 0F5F5H, 0F5F4H, 0F4F4H, 0F3F3H, 0F3F2H, 0F2F2H, 0F1F1H, 0F1F0H
03A2	F5F4		
03A4	F4F4		
03A6	F3F3		
03A8	F3F2		

LOC	OBJ	LINE	SOURCE STATEMENT
03AA	F2F2		
03AC	F1F1		
03AE	F1F0		
03B0	F0F0	145	DW 0F0F0H, 0EFEF0H, 0EFEEH, 0EEEEH, 0EDEDH, 0ECECH, 0ECEBH, 0EBEBH
03B2	EFEF		
03B4	EFEE		
03B6	EEEE		
03B8	EDED		
03BA	ECEC		
03BC	ECCE		
03BE	EBEB		
03C0	EAEA	146	DW 0EAEAH, 0E9E9H, 0E9E8H, 0E8E8H, 0E7E7H, 0E7E6H, 0E6E5H, 0E4E4H
03C2	E9E9		
03C4	E9E8		
03C6	E8E8		
03C8	E7E7		
03CA	E7E6		
03CC	E6E5		
03CE	E4E4		
03D0	E3E3	147	DW 0E3E3H, 0E2E2H, 0E1E1H, 0E0E0H, 0E0DFH, 0EDEEH, 0DDDDH, 0DCDEH
03D2	E2E2		
03D4	E1E1		
03D6	E0E0		
03D8	E0DF		
03DA	DEDE		
03DC	DDDD		
03DE	DCDE		
03E0	DBDA	148	DW 0DBDAH, 0D9D9H, 0D8D7H, 0D7D6H, 0D5D4H, 0D4D3H, 0D2D1H, 0DOCFH
03E2	D9D9		
03E4	D8D7		
03E6	D7D6		
03E8	D5D4		
03EA	D4D3		
03EC	D2D1		
03EE	DOCF		
03F0	CECD	149	DW 0CECDH, 0CCCAH, 0C9C8H, 0C6C4H, 0C2C0H, 0EBEBH, 0B8B3H, 0AB81H
03F2	CCCA		
03F4	C9C8		
03F6	C6C4		
03F8	C2C0		
03FA	EBEB		
03FC	B8B3		
03FE	AB81		
		150	END

USER SYMBOLS

INIT 0053 INRNG 002B NEXT 0005 NOCHG 0020 SUB1D 0017 SUB8

ASSEMBLY COMPLETE, 1 ERROR ( 10 )

Appx 'G'

ISIS-II 8080/8085 MACRO ASSEMBLER, V4.1

MODULE PAGE 1

LOC	OBJ	LINE	SOURCE STATEMENT
6000		1	ORG 6000H
6000	20202020	2	M1: DB / WELCOME TO TROPOSCATTER SIMULATOR*/
6004	57454C43		
6008	4F4D4520		
600C	544F2054		
6010	524F504F		
6014	53434154		
6018	54455220		
601C	53494D55		
6020	4C41544F		
6024	522A		
6026	20202020	3	M2: DB / -----*/
602A	2D2D2D2D		
602E	2D2D2D2D		
6032	2D2D2D2D		
6036	2D2D2D2D		
603A	2D2D2D2D		
603E	2D2D2D2D		
6042	2D2D2D2D		
6046	2D2D2D2D		
604A	2D2A		
604C	20202020	4	M3: DB / INTERACTION PHASE: -, OAH, ODH
6050	494E5445		
6054	52414354		
6058	494F4E20		
605C	50484153		
6060	453A2D		
6063	0A		
6064	0D		
6065	20202020	5	DB / SIMULATION WILL START AFTER INITIALISATION. -, OAH,
6069	53494D55		
606D	4C415449		
6071	4F4E2057		
6075	494C4C20		
6079	53544152		
607D	54204146		
6081	54455220		
6085	494E4954		
6089	49414C49		
608D	53415449		
6091	4F4E2E20		
6095	0A		
6096	0D		
6097	20202020	6	DB / REQUIRED PARAMETERS ARE : - -, OAH, ODH
609B	52455155		
609F	49524544		
60A3	20504152		
60A7	414D4554		
60AB	45525320		
60AF	41524520		
60B3	3A2D20		
60B6	0A		
60B7	0D		
60B8	20202020	7	DB / (1) NUMBER OF TAPS ( SPECIFY ANY NUMBER -, OAH, ODH

LOC	OBJ	LINE	SOURCE STATEMENT
60BC	28312920		
60C0	4E554D42		
60C4	4552204F		
60C8	46205441		
60CC	50532028		
60D0	20535045		
60D4	43494659		
60D8	20414E59		
60DC	204E554D		
60E0	424552		
60E3	0A		
60E4	0D		
60E5	20202020	8 DB	BETWEEN 01 AND 05). /, 0AH, 0DH
60E9	20202042		
60ED	45545745		
60F1	454E2030		
60F5	3120414E		
60F9	44203035		
60FD	292E		
60FF	0A		
6100	0D		
6101	20202020	9 DB	(2) SAMPLING INTERVAL PER TAP IN MILLISEC /, 0AH, 0DH
6105	28322920		
6109	53414D50		
610D	4C494E47		
6111	20494E54		
6115	45525641		
6119	4C205045		
611D	52205441		
6121	5020494E		
6125	204D494C		
6129	4C495345		
612D	4320		
612F	0A		
6130	0D		
6131	20202020	10 DB	( SPECIFY ANY VALUE BETWEEN 01 AND 16 ). /, 0AH, 0DH
6135	20202028		
6139	20535045		
613D	43494659		
6141	20414E59		
6145	2056414C		
6149	55452042		
614D	45545745		
6151	454E2030		
6155	3120414E		
6159	44203136		
615D	20292E		
6160	0A		
6161	0D		
6162	20202020	11 DB	(3) BANDWIDTH REQUIRED ON EACH TAP /, 0AH, 0DH
6166	28332920		
616A	42414E44		
616E	57494454		
6172	48205245		
6176	51554952		

LOC	OBJ	LINE	SOURCE STATEMENT
617A	4544204F		
617E	4E204541		
6182	43482054		
6186	4150		
6188	0A		
6189	0D		
618A	20202020	12 DB /	IN HZ (SPECIFY ANY NUMBER BETWEEN 01 AND 16)...
618E	20202020		
6192	494E2048		
6196	5A202853		
619A	50454349		
619E	46592041		
61A2	4E59204E		
61A6	554D4245		
61AA	52204245		
61AE	54574545		
61B2	4E202030		
61B6	3120414E		
61BA	44203136		
61BE	292E		
61C0	0A		
61C1	0D		
61C2	0D		
61C3	20202020	13 DB /	NOW TYPE IN PARAMETERS IN THE FOLLOWING /, 0AH, 0DH
61C7	204E4F57		
61CB	20545950		
61CF	4520494E		
61D3	20504152		
61D7	414D4554		
61DB	45525320		
61DF	494E2054		
61E3	48452046		
61E7	4F4C4C4F		
61EB	57494E47		
61EF	20		
61F0	0A		
61F1	0D		
61F2	0D		
61F3	20202020	14 DB /	FORMAT :- /, 0AH, 0DH
61F7	20464F52		
61FB	4D415420		
61FF	3A2D2020		
6203	20		
6204	0A		
6205	0D		
6206	20202020	15 DB /	TAPS SAMP: INT BWCH1 BWCH2 BWCH3 BWCH4 BWCH5 CR/, C
620A	20544150		
620E	53205341		
6212	4D503A49		
6216	4E542042		
621A	57434831		
621E	20425743		
6222	48322042		
6226	57434833		
622A	20425743		

LOC	OBJ	LINE	SOURCE STATEMENT
622E	48342042		
6232	57434835		
6236	204352		
6239	0A		
623A	0D		
623B	20202020	16 DB /	(FOR EX. 02 01 10 08 04 09 16 CR)* /
623F	20202020		
6243	20202028		
6247	464F5220		
624B	45582E20		
624F	30322030		
6253	31203130		
6257	20303820		
625B	30342030		
625F	39203136		
6263	20435229		
6267	2A		
6268	20202045	17 M4: DB /	ERROR 1 :- SPECIFIED TAPS OUT OF RANGE. * /
626C	52524F52		
6270	2031203A		
6274	2D205350		
6278	45434946		
627C	49454420		
6280	54415053		
6284	204F5554		
6288	204F4620		
628C	52414E47		
6290	452E2A		
6293	20202045	18 M5: DB /	ERROR 2 :- SPECIFIED SAMPLING TIME OUT OF RANGE. * /
6297	52524F52		
629B	2032203A		
629F	2D205350		
62A3	45434946		
62A7	49454420		
62AB	53414D50		
62AF	4C494E47		
62B3	2054494D		
62B7	45204F55		
62BB	54204F46		
62BF	2052414E		
62C3	47452E2A		
62C7	20202045	19 M6: DB /	ERROR 3 :- SPECIFIED BAND WIDTH OUT OF RANGE. * /
62CB	52524F52		
62CF	2033203A		
62D3	2D205350		
62D7	45434946		
62DB	49454420		
62DF	42414E44		
62E3	20574944		
62E7	5448204F		
62EB	5554204F		
62EF	46205241		
62F3	4E47452E		
62F7	2A		
62F8	20202020	20 M7: DB /	SIMULATOR ON* /



LOC	OBJ	LINE	SOURCE STATEMENT
62FC	20202020		
6300	20205349		
6304	4D554C41		
6308	544F5220		
630C	4F4E2A		
630F	20202020	21 M8:DB /	(1) ESC TO END SIMULATION. /, 0AH, 0DH
6313	20202028		
6317	31292045		
631B	53432054		
631F	4F20454E		
6323	44205349		
6327	4D554C41		
632B	54494F4E		
632F	2E		
6330	0A		
6331	0D		
6332	20202020	22 DB /	(2) ANY OTHER KEY TO ALTER PARAMETERS. */
6336	20202028		
633A	32292041		
633E	4E59204F		
6342	54484552		
6346	204B4559		
634A	20544F20		
634E	414C5445		
6352	52205041		
6356	52414D45		
635A	54455253		
635E	2E2A		
6360	20202020	23 M9:DB /	REINITIALISATION PHASE ( SIMULATION TERMIN
6364	20205245		
6368	494E4954		
636C	49414C49		
6370	53415449		
6374	4F4E2050		
6378	48415345		
637C	20202020		
6380	28205349		
6384	4D554C41		
6388	54494F4E		
638C	20544552		
6390	4D494E41		
6394	54454420		
6398	292E2A		
639B	20202020	24 M10:DB /	END OF SIMULATON */
639F	20202020		
63A3	20202020		
63A7	20202020		
63AB	454E4420		
63AF	4F462053		
63B3	494D554C		
63B7	41544F4E		
63BB	202A		
63BD	20202020	25 M11:DB /	ENSURE (JMP 651F) INSTR A1 5812 FOR RST 6.5
63C1	454E5355		
63C5	52452028		

LOC	OBJ	LINE	SOURCE STATEMENT
63C9	4A4D5020		
63CD	36353146		
63D1	29202049		
63D5	4E535452		
63D9	20415420		
63DD	35383132		
63E1	20464F52		
63E5	20525354		
63E9	20362E35		
63ED	20494E54		
63F1	522E2A		
63F4	20202020	26	M20: DB / DATA ACKNOWLEDGED*/
63F8	20202020		
63FC	20202020		
6400	20202020		
6404	44415441		
6408	2041434B		
640C	4E4F574C		
6410	45444745		
6414	442A		
6416	3E00	27	BINRY: MVI A, 00H ; BCD TO BINARY CONVERTER
6418	320056	28	STA 5600H
641B	320156	29	STA 5601H ; REQUIRES THE BCD NUMBER IN 5602
641E	110056	30	LXI D, 5600H
6421	3A0256	31	LDA 5602H
6424	E60F	32	ANI 0FH
6426	320356	33	STA 5603H
6429	3A0256	34	LDA 5602H
642C	E6F0	35	ANI 0F0H
642E	0F	36	RRC
642F	0F	37	RRC
6430	0F	38	RRC
6431	0F	39	RRC
6432	E60F	40	ANI 0FH ; GET TO LEAST SIGNIFICANT BYTE
6434	320256	41	STA 5602H
6437	CD7600	42	CALL 0076H ; BCDTB SUBROUTINE
643A	7D	43	MOV A, L
643B	320256	44	STA 5602H ; HEX RESULT IN 5602H
643E	C9	45	RET
643F	3E92	46	TRW: MVI A, 92H ; TRW 1008 INITIALISATION
6441	D3CC	47	OUT 0CCH
6443	3E00	48	MVI A, 00H
6445	D3C0	49	OUT 0C0H
6447	D3C4	50	OUT 0C4H
6449	D3D4	51	OUT 0D4H
644B	D3D8	52	OUT 0D8H
644D	D3DC	53	OUT 0DCH
644F	C9	54	RET
6450	3D	55	ALPHA: DCR A ; READS VALUE OF PARAMETER ALPHA
6451	E60F	56	ANI 0FH ; FM ROM AREA 6F00H TO 6FFFH
6453	80	57	ADD B ; BY FORMING ITS ADDR L FM
6454	6F	58	MOV L, A ; SAMPLING TIME AND BW REQD
6455	7E	59	MOV A, M ; INITIALISES LOC 5711H TO 571AH
6456	EB	60	XCHG ; WITH VALUE OF ALPHA
6457	77	61	MOV M, A

LOC	OBJ	LINE	SOURCE STATEMENT
6458	23	62	INX H
6459	77	63	MOV M, A ; SAME ALPHA FOR TWO CHS (SAME TAP
645A	23	64	INX H
645B	EB	65	XCHG ; THE ABOVE SUB REQUIRES BW IN ACC
645C	C9	66	RET ; 6F00H =(HL) AND 5711H =(DE)
645D	111157	67	INIT1: LXI D, 5711H ; SUB TO CALCULATE VALUE OF ALPHA
6460	21006F	68	LXI H, 6F00H
6463	3A0157	69	LDA 5701H ; SAMPLING TIME
6466	3D	70	DCR A
6467	07	71	RLC
6468	07	72	RLC
6469	07	73	RLC
646A	07	74	RLC
646B	E6F0	75	ANI 0F0H ; FORM ADDR L UPPER NIBBLE
646D	47	76	MOV B, A
646E	3A0257	77	LDA 5702H ; BW TAP 1
6471	CD5064	78	CALL ALPHA
6474	3A0357	79	LDA 5703H ; BW TAP 2
6477	CD5064	80	CALL ALPHA
647A	3A0457	81	LDA 5704H ; BW TAP 3
647D	CD5064	82	CALL ALPHA
6480	3A0557	83	LDA 5705H ; BW TAP 4
6483	CD5064	84	CALL ALPHA
6486	3A0657	85	LDA 5706H ; BW TAP 5
6489	CD5064	86	CALL ALPHA
648C	060A	87	MVI B, 0AH
648E	3E00	88	MVI A, 00H
6490	212157	89	LXI H, 5721H
6493	77	90	IT: MOV M, A ; THIS INITIALISES MEM LOC
6494	23	91	INX H ; 5721H TO 572AH =00H
6495	05	92	DCR B
6496	C29364	93	JNZ IT
6499	C9	94	RET
649A	01F700	95	INIT2: LXI B, 00F7H ; (BC)=247
649D	110000	96	LXI D, 0000H
64A0	3A0157	97	LDA 5701H ; SAMPLING TIME
64A3	5F	98	MOV E, A
64A4	CD7C00	99	CALL 007CH ; EMUL SUBROUTINE
64A7	4D	100	MOV C, L ; (DE) * (BC) = (DEHL)
64A8	44	101	MOV B, H
64A9	EB	102	XCHG
64AA	110700	103	LXI D, 0007H ; DELAY=([7.2 * (BC)] + 9) MICROSE
64AD	CD7F00	104	CALL 007FH ; BDIV SUB
64B0	223057	105	SHLD 5730H ; VALUE OF DELAY STORED
64B3	C9	106	RET
64B4	3A0057	107	INIT3: LDA 5700H ; TAPS
64B7	37	108	STC
64B8	3F	109	CMC
64B9	07	110	RLC
64BA	320757	111	STA 5707H ; NUMBER OF PROCESSES = TAPS * 2
64BD	320857	112	STA 5708H
64C0	3E80	113	MVI A, 80H ; INITIALISE 8255S, TRW 1008 ETC
64C2	D3B3	114	OUT 0B3H ; 8255 NO. , 1
64C4	D3B7	115	OUT 0B7H ; NO. , 2
64C6	D3BB	116	OUT 0BBH ; NO. , 3

LOC	OBJ	LINE	SOURCE STATEMENT
64C8	D3BF	117	OUT 0BFH ; NO. , 4
64CA	CD3F64	118	CALL TRW
64CD	3E1D	119	MVI A,1DH
64CF	30	120	SIM ; MASK RST 7.5, INTR AND UNMASK RST
64D0	212057	121	LXI H,5720H ; ADDR OF PREVIOUS SAMPLES
64D3	3A0757	122	LDA 5707H ; POINT TO HIGHEST SAMPLE ADDR
64D6	85	123	ADD L ; PREVIOUS SAMPLES STORED AT 5721H
64D7	6F	124	MOV L,A
64D8	220957	125	SHLD 5709H ; ADDR STORED AT 5709H,0AH
64DB	211057	126	LXI H,5710H
64DE	3A0757	127	LDA 5707H ; ALPHA OF HIGHEST ADDR
64E1	85	128	ADD L ; ALPHAS AVAILABLE AT 5711H TO 571
64E2	6F	129	MOV L,A
64E3	220B57	130	SHLD 570BH ; ADDR STORED AT 570BH,0CH
64E6	3A0757	131	LDA 5707H
64E9	21006E	132	LXI H,6E00H ; ROM AREA FOR ALPHA
64EC	3D	133	AGAIN: DCR A
64ED	CAF664	134	JZ DAK ; FIX JMP ADDR FOR DACS
64F0	2C	135	INR L
64F1	2C	136	INR L
64F2	2C	137	INR L
64F3	C3EC64	138	JMP AGAIN
64F6	220D57	139	DAK: SHLD 570DH ; JMP ADDR FOR HIGHEST DAC
64F9	7D	140	MOV A,L
64FA	320F57	141	STA 570FH ; ALSO STORED AT 570FH
64FD	C9	142	RET
64FE	78	143	DAC1: MOV A,B ; GET SAMPLE
64FF	D3B2	144	OUT 0B2H ; SEND TO DAC 1
6501	C3BF65	145	JMP BAK ; GO BACK TO MAIN PGME
6504	78	146	DAC2: MOV A,B
6505	D3B4	147	OUT 0B4H ; DAC 2
6507	C3BF65	148	JMP BAK
650A	78	149	DAC3: MOV A,B
650B	D3B5	150	OUT 0B5H
650D	C3BF65	151	JMP BAK
6510	78	152	DAC4: MOV A,B
6511	D3B6	153	OUT 0B6H
6513	C3BF65	154	JMP BAK
6516	78	155	DAC5: MOV A,B
6517	D3B8	156	OUT 0B8H
6519	C3BF65	157	JMP BAK
651C	78	158	DAC6: MOV A,B
651D	D3B9	159	OUT 0B9H
651F	C3BF65	160	JMP BAK
6522	78	161	DAC7: MOV A,B
6523	D3BA	162	OUT 0BAH
6525	C3BF65	163	JMP BAK
6528	78	164	DAC8: MOV A,B
6529	D3BC	165	OUT 0BCH
652B	C3BF65	166	JMP BAK
652E	78	167	DAC9: MOV A,B
652F	D3BD	168	OUT 0BDH
6531	C3BF65	169	JMP BAK
6534	78	170	DAC10: MOV A,B
6535	D3BE	171	OUT 0BEH

LOC	OBJ	LINE	SOURCE STATEMENT
6537	C3BF65	172	JMP BAK
653A	3E98	173	ISR: MVI A, 98H
653C	D3CC	174	OUT 0CCH ; 2'S COMP MULT MODE OF 1008
653E	2A0957	175	LHLD 5709H ; ADDR OF CURRENT SAMPLE
6541	223257	176	SHLD 5732H ; SAVE CURRENT ADDR
6544	7E	177	MOV A, M
6545	D3C0	178	OUT 0C0H ; X INPUT OF TRW
6547	7D	179	MOV A, L
6548	E60F	180	ANI 0FH ; REMOVE UPPER NIBBLE
654A	3D	181	DCR A
654B	C25165	182	JNZ NOALX
654E	3A0757	183	LDA 5707H ; RESET TO HIGHEST ADDR
6551	C620	184	NOALX: ADI 20H
6553	6F	185	MOV L, A ; ADDR OF NEXT SAMPLE
6554	220957	186	SHLD 5709H ; ADDR OF ALPHA
6557	2A0B57	187	LHLD 570BH
655A	7E	188	MOV A, M
655B	D3C4	189	OUT 0C4H ; Y INPUT
655D	7D	190	MOV A, L
655E	E60F	191	ANI 0FH
6560	3D	192	DCR A
6561	C26765	193	JNZ NOALY
6564	3A0757	194	LDA 5707H ; RESET ADDR TO HIGHEST VALUE
6567	C610	195	NOALY: ADI 10H
6569	6F	196	MOV L, A
656A	220B57	197	SHLD 570BH
656D	DBD4	198	IN 0D4H ; READ TRW LSP
656F	E680	199	ANI 80H
6571	C27965	200	JNZ NZERO
6574	1E00	201	MVI E, 00H
6576	C37B65	202	JMP ZERO
6579	1E01	203	NZERO: MVI E, 01H
657B	3E10	204	ZERO: MVI A, 10H
657D	D3CC	205	OUT 0CCH ; DISABLE TRW
657F	DBD8	206	IN 0D8H ; READ MSP
6581	07	207	RLC
6582	E6FE	208	ANI 0FEH ; THIS IS FOR DESIMAL POINT ADJUSTMEI
6584	83	209	ADD E
6585	5F	210	MOV E, A
6586	CD3F64	211	CALL TRW
6589	DBD0	212	IN 0D0H ; READ 8741
658B	D3D4	213	OUT 0D4H ; PRELOAD LTP O/P BUFFER
658D	3E9A	214	MVI A, 9AH ; 2'S COMP, ACC & MULT MODE
658F	D3CC	215	OUT 0CCH
6591	7B	216	MOV A, E
6592	D3C0	217	OUT 0C0H ; X INPUT
6594	3E01	218	MVI A, 01H ; Y INPUT
6596	D3C4	219	OUT 0C4H
6598	DBD4	220	IN 0D4H ; READ LTP OF TRW
659A	47	221	MOV B, A
659B	E680	222	ANI 80H
659D	CAA765	223	JZ POSVE
65A0	78	224	MOV A, B
65A1	E67F	225	ANI 7FH
65A3	2F	226	CMA

LOC	OBJ	LINE	SOURCE STATEMENT
65A4	C601	227	ADI 01H
65A6	47	228	MOV B, A ; RNG SAMPLE TO BE OUTPUT ON DACS SEQ
65A7	2A0D57	229	POSV: LHL 570DH ; ADDR OF DAC VIA JMP INSTR
65AA	EB	230	XCHG
65AB	62	231	MOV H, D
65AC	6B	232	MOV L, E
65AD	2D	233	DCR L
65AE	2D	234	DCR L
65AF	2D	235	DCR L
65B0	C2BA65	236	JNZ STORE ; NEXT ADDR IS BEING FORMED
65B3	21006E	237	LXI H, 6E00H
65B6	3A0F57	238	LDA 570FH
65B9	6F	239	MOV L, A
65BA	220D57	240	STORE: SHLD 570DH
65BD	EB	241	XCHG
65BE	E9	242	PCHL ; EQUIVALENT TO A JMP INSTR
65BF	2A3257	243	BAK: LHL 5732H ; REENTRY POINT
65C2	70	244	MOV M, B
65C3	2A3057	245	LHL 5730H ; FETCH VALUE OF DELAY
65C6	44	246	MOV B, H
65C7	4D	247	MOV C, L
65C8	CD6A00	248	CALL 006AH ; DELAY SUBROUTINE
65CB	DBF9	249	IN OF9H
65CD	C9	250	RET ; DELAY= ((BC)*7.2+9) MICROSECS
65CE	010060	251	START: LXI B, M1
65D1	CD5E00	252	CALL 005EH ; PNTMS
65D4	CD4600	253	CALL 0046H ; CRLF
65D7	012660	254	LXI B, M2
65DA	CD5E00	255	CALL 005EH
65DD	CD4600	256	CALL 0046H
65E0	CD4600	257	CALL 0046H
65E3	CD4600	258	CALL 0046H
65E6	01BD63	259	LXI B, M11
65E9	CD5E00	260	CALL 005EH
65EC	CD4600	261	CALL 0046H
65EF	CD4600	262	CALL 0046H
65F2	CD4600	263	CALL 0046H
65F5	014C60	264	MS3: LXI B, M3
65F8	CD5E00	265	CALL 005EH
65FB	CD4600	266	CALL 0046H
65FE	210057	267	LXI H, 5700H
6601	CD6100	268	CALL 0061H ; HXCHR
6604	3E0C	269	MVI A, 0CH
6606	CD4300	270	CALL 0043H
6609	01F463	271	LXI B, M20
660C	CD5E00	272	CALL 005EH
660F	CD4600	273	CALL 0046H
6612	CD4600	274	CALL 0046H
6615	01FF0F	275	LXI B, 0FFFH
6618	CD6A00	276	CALL 006AH ; DELAY
661B	3A0057	277	LDA 5700H
661E	FE06	278	CPI 06H ; CHECK IF IN RANGE
6620	D22866	279	JNC MSG4
6623	FE01	280	CPI 01H
6625	D23C66	281	JNC NEXT

LOC	OBJ	LINE	SOURCE STATEMENT
6628	3E0C	282	MSG4: MVI A, 0CH
662A	CD4300	283	CALL 0043H ; PRINT FORM FEED
662D	016862	284	LXI B, M4
6630	CD5E00	285	CALL 005EH
6633	CD4600	286	CALL 0046H
6636	CD4600	287	CALL 0046H
6639	C3F565	288	JMP MS3 ; ASK FOR DATA AGAIN
663C	3A0157	289	NEXT: LDA 5701H ; CONVERT ALL BCDTB
663F	320256	290	STA 5602H
6642	CD1664	291	CALL BINRY
6645	3A0256	292	LDA 5602H
6648	320157	293	STA 5701H
664B	FE11	294	CPI 11H
664D	DA6466	295	JC OK1
6650	3E0C	296	MVI A, 0CH
6652	CD4300	297	CALL 0043H ; FORM FEED
6655	019362	298	LXI B, M5
6658	CD5E00	299	CALL 005EH
665B	CD4600	300	CALL 0046H
665E	CD4600	301	CALL 0046H
6661	C3F565	302	JMP MS3 ; ASK FOR DATA AGAIN
6664	3A0257	303	OK1: LDA 5702H
6667	320256	304	STA 5602H
666A	CD1664	305	CALL BINRY
666D	3A0256	306	LDA 5602H
6670	320257	307	STA 5702H
6673	FE11	308	CPI 11H
6675	DA8C66	309	JC OK2
6678	3E0C	310	MSG: MVI A, 0CH
667A	CD4300	311	CALL 0043H ; FORM FEED
667D	01C762	312	LXI B, M6
6680	CD5E00	313	CALL 005EH
6683	CD4600	314	CALL 0046H
6686	CD4600	315	CALL 0046H
6689	C3F565	316	JMP MS3
668C	3A0057	317	OK2: LDA 5700H ; NO. , OF TAPS
668F	3D	318	DCR A
6690	CAFB66	319	JZ GOON
6693	3A0357	320	LDA 5703H
6696	320256	321	STA 5602H
6699	CD1664	322	CALL BINRY
669C	3A0256	323	LDA 5602H
669F	320357	324	STA 5703H
66A2	FE11	325	CPI 11H
66A4	D27866	326	JNC MSG
66A7	3A0057	327	LDA 5700H
66AA	FE02	328	CPI 02H
66AC	CAFB66	329	JZ GOON
66AF	3A0457	330	LDA 5704H
66B2	320256	331	STA 5602H
66B5	CD1664	332	CALL BINRY
66B8	3A0256	333	LDA 5602H
66BB	320457	334	STA 5704H
66BE	FE11	335	CPI 11H
66C0	D27866	336	JNC MSG

LOC	OBJ	LINE	SOURCE STATEMENT
66C3	3A0057	337	LDA 5700H
66C6	FE03	338	CPI 03H
66C8	CAFB66	339	JZ 000N
66CB	3A0557	340	LDA 5705H
66CE	320256	341	STA 5602H
66D1	CD1664	342	CALL BINRY
66D4	3A0256	343	LDA 5602H
66D7	320557	344	STA 5705H
66DA	FE11	345	CPI 11H
66DC	D27866	346	JNC MSG
66DF	3A0057	347	LDA 5700H
66E2	FE04	348	CPI 04H
66E4	CAFB66	349	JZ 000N
66E7	3A0657	350	LDA 5706H
66EA	320256	351	STA 5602H
66ED	CD1664	352	CALL BINRY
66F0	3A0256	353	LDA 5602H
66F3	320657	354	STA 5706H
66F6	FE11	355	CPI 11H
66F8	D27866	356	JNC MSG
66FB	CD5D64	357	000N:CALL INIT1
66FE	CD9A64	358	CALL INIT2
6701	CDB464	359	CALL INIT3
6704	3E0C	360	MVI A,0CH
6706	CD4300	361	CALL 0043H
6709	01F862	362	LXI B,M7
670C	CD5E00	363	CALL 005EH
670F	CD4600	364	CALL 0046H
6712	CD4600	365	CALL 0046H
6715	CD4600	366	CALL 0046H
6718	010F63	367	LXI B,M8
671B	CD5E00	368	CALL 005EH
671E	CD4600	369	CALL 0046H
6721	DBF8	370	IN 0F8H
6723	FB	371	WAIT:EI
6724	DBF9	372	IN 0F9H ; READ STATUS OR 8251 OF KEYBOARD
6726	E602	373	ANI 02H ; WAIT FOR RST 6.5
6728	CA2367	374	JZ WAIT
672B	F3	375	DI
672C	DBF8	376	IN 0F8H ; READ KEYBOARD
672E	FE1B	377	CPI 1BH ; IS ESC KEY PRESSED
6730	CA4A67	378	JZ FINIS
6733	3E0C	379	MVI A,0CH
6735	CD4300	380	CALL 0043H
6738	016063	381	LXI B,M9
673B	CD5E00	382	CALL 005EH
673E	CD4600	383	CALL 0046H
6741	CD4600	384	CALL 0046H
6744	CD4600	385	CALL 0046H
6747	C3F565	386	JMP MS3
674A	3E0C	387	FINIS:MVI A,0CH
674C	CD4300	388	CALL 0043H
674F	019B63	389	LXI B,M10
6752	CD5E00	390	CALL 005EH
6755	CD4600	391	CALL 0046H



LOC	OBJ	LINE	SOURCE STATEMENT
6758	CD4600	392	CALL 0046H
675B	CD4600	393	CALL 0046H
675E	CF	394	RST 1
675F	3E90	395	MVI A, 90H
6761	D3CC	396	OUT 0CCH ; CHECK RNG DATA SUBROUTINE
6763	DBD0	397	IN 0D0H
6765	47	398	MOV B, A
6766	E680	399	ANI 80H
6768	CA7267	400	JZ P1P
676B	78	401	MOV A, B
676C	E67F	402	ANI 7FH
676E	2F	403	CMA
676F	C601	404	ADI 01H
6771	47	405	MOV B, A
6772	78	406	P1P: MOV A, B
6773	D3B8	407	OUT 0B8H
6775	2A3057	408	LHLD 5730H
6778	44	409	MOV B, H
6779	4D	410	MOV C, L
677A	CD6A00	411	CALL 006AH
677D	3E00	412	MVI A, 00H
677F	C9	413	RET
6780	3E90	414	MVI A, 90H
6782	D3CC	415	OUT 0CCH
6784	DBD0	416	IN 0D0H
6786	CD5800	417	CALL 0058H
6789	2A3057	418	LHLD 5730H
678C	44	419	MOV B, H
678D	4D	420	MOV C, L
678E	CD6A00	421	CALL 006AH
6791	3E00	422	MVI A, 00H
6793	C9	423	RET
6794	2A0957	424	RESP0: LHLD 5709H
6797	7E	425	MOV A, M ; GET PREVIOUS SAMPLE
6798	323457	426	STA 5734H
679B	2A0B57	427	LHLD 570BH
679E	7E	428	MOV A, M ; GET COEFFICIENT
679F	323557	429	STA 5735H
67A2	3E98	430	MVI A, 98H
67A4	D3CC	431	OUT 0CCH
67A6	3A3457	432	LDA 5734H
67A9	D3C0	433	OUT 0C0H
67AB	3A3557	434	LDA 5735H
67AE	D3C4	435	OUT 0C4H
67B0	DBD4	436	IN 0D4H
67B2	E680	437	ANI 80H
67B4	C2BC67	438	JNZ NZR
67B7	1E00	439	MVI E, 00H
67B9	C3BE67	440	JMP ZER
67BC	1E10	441	NZR: MVI E, 10H
67BE	3E10	442	ZER: MVI A, 10H
67C0	D3CC	443	OUT 0CCH
67C2	07	444	RLC
67C3	E6FE	445	ANI 0FEH
67C5	83	446	ADD E

LOC	OBJ	LINE	SOURCE STATEMENT
67C6	5F	447	MOV E, A
67C7	CD3F64	448	CALL TRW
67CA	DEDO	449	IN OD0H
67CC	3A3657	450	LDA 5736H
67CF	D3D4	451	OUT OD4H
67D1	3E9A	452	MVI A, 9AH
67D3	D3CC	453	OUT OCCH
67D5	7B	454	MOV A, E
67D6	D3C0	455	OUT OC0H
67D8	3E01	456	MVI A, 01H
67DA	D3C4	457	OUT OC4H
67DC	DEB4	458	IN OD4H
67DE	323457	459	STA 5734H
67E1	213457	460	LXI H, 5734H
67E4	2A0957	461	LHLD 5709H
67E7	77	462	MOV M, A
67E8	47	463	MOV B, A
67E9	E680	464	ANI 80H
67EB	CAF567	465	JZ P3P
67EE	78	466	MOV A, B
67EF	E67F	467	ANI 7FH
67F1	2F	468	CMA
67F2	C601	469	ADI 01H
67F4	47	470	MOV B, A
67F5	78	471	P3P: MOV A, B
67F6	D3B8	472	OUT OB8H
67F8	CD5800	473	CALL 0058H
67FB	CD4900	474	CALL 0049H
67FE	2A3057	475	LHLD 5730H
6801	44	476	MOV B, H
6802	4D	477	MOV C, L
6803	CD6A00	478	CALL 006AH
6806	3E00	479	MVI A, 00H
6808	C9	480	RET
6DFD		481	ORG 6DFDH
6DFD	CF	482	RST 1
6DFE	CF	483	RST 1
6DFF	CF	484	RST 1
6E00	C3FE64	485	JMP DAC1
6E03	C30465	486	JMP DAC2
6E06	C30A65	487	JMP DAC3
6E09	C31065	488	JMP DAC4
6E0C	C31665	489	JMP DAC5
6E0F	C31C65	490	JMP DAC6
6E12	C32265	491	JMP DAC7
6E15	C32865	492	JMP DAC8
6E18	C32E65	493	JMP DAC9
6E1B	C33465	494	JMP DAC10
6E1E	CF	495	RST 1
6E1F	CF	496	RST 1
6E20	CF	497	RST 1
6E21	CF	498	RST 1
6F00		499	ORG 6F00H ; ALPHA VALUES
6F00	7F7E	500	DW 7E7FH, 7D7EH, 7B7CH, 7A7BH, 7879H, 7777H, 7576H, 7475H
6F02	7E7D		

LOC	OBJ	LINE	SOURCE STATEMENT
6F04	7C7B		
6F06	7B7A		
6F08	7978		
6F0A	7777		
6F0C	7675		
6F0E	7574		
6F10	7E7D	501	DW 7D7EH, 7A7BH, 7778H, 7475H, 7173H, 6E6FH, 6B6CH, 696BH
6F12	7B7A		
6F14	7877		
6F16	7574		
6F18	7371		
6F1A	6F6E		
6F1C	6C6B		
6F1E	6B69		
6F20	7D7B	502	DW 7B7DH, 777AH, 7275H, 6E70H, 6A6DH, 6667H, 6263H, 5F62H
6F22	7A77		
6F24	7572		
6F26	706E		
6F28	6D6A		
6F2A	6766		
6F2C	6362		
6F2E	625F		
6F30	7D7A	503	DW 7A7DH, 7477H, 6E71H, 696CH, 6466H, 5E60H, 595CH, 5659H
6F32	7774		
6F34	716E		
6F36	6C69		
6F38	6664		
6F3A	605E		
6F3C	5C59		
6F3E	5956		
6F40	7C78	504	DW 787CH, 7175H, 6A6EH, 6467H, 5E61H, 5759H, 5154H, 4E51H
6F42	7571		
6F44	6E6A		
6F46	6764		
6F48	615E		
6F4A	5957		
6F4C	5451		
6F4E	514E		
6F50	7B77	505	DW 777BH, 6E72H, 666AH, 5F63H, 585CH, 5154H, 4A4EH, 464AH
6F52	726E		
6F54	6A66		
6F56	635F		
6F58	5C58		
6F5A	5451		
6F5C	4E4A		
6F5E	4A46		
6F60	7A75	506	DW 757AH, 6B70H, 6267H, 5A5EH, 5357H, 4B4EH, 4447H, 4044H
6F62	706B		
6F64	6762		
6F66	5E5A		
6F68	5753		
6F6A	4E4B		
6F6C	4744		
6F6E	4440		
6F70	7A74	507	DW 747AH, 696EH, 5E64H, 565AH, 4E52H, 4548H, 3E42H, 393EH

LOC	OBJ	LINE	SOURCE STATEMENT
6F72	6E69		
6F74	645E		
6F76	5A56		
6F78	524E		
6F7A	4845		
6F7C	423E		
6F7E	3E39		
6F80	7972	508	DW 7279H, 666CH, 5B61H, 5257H, 494EH, 4043H, 393DH, 3439H
6F82	6C66		
6F84	615B		
6F86	5752		
6F88	4E49		
6F8A	4340		
6F8C	3D39		
6F8E	3934		
6F90	7871	509	DW 7178H, 646AH, 575EH, 4E53H, 4449H, 3B3FH, 3438H, 2F23H
6F92	6A64		
6F94	5E57		
6F96	534E		
6F98	4944		
6F9A	3F3B		
6F9C	3834		
6F9E	232F		
6FA0	7770	510	DW 7077H, 6168H, 545BH, 4A4FH, 4045H, 373AH, 2F33H, 2B2FH
6FA2	6861		
6FA4	5B54		
6FA6	4F4A		
6FA8	4540		
6FAA	3A37		
6FAC	332F		
6FAE	2F2B		
6FB0	776E	511	DW 6E77H, 5F66H, 5158H, 464CH, 3C42H, 3336H, 2B2FH, 272BH
6FB2	665F		
6FB4	5851		
6FB6	4C46		
6FB8	423C		
6FBA	3633		
6FBC	2F2B		
6FBE	2B27		
6FC0	766D	512	DW 6D76H, 5D64H, 4E55H, 4349H, 393EH, 2F33H, 272BH, 2327H
6FC2	645D		
6FC4	554E		
6FC6	4943		
6FC8	3E39		
6FCA	332F		
6FCC	2B27		
6FCE	2723		
6FD0	756B	513	DW 6B75H, 5A62H, 4B53H, 3F46H, 353BH, 2C2FH, 2428H, 1F24H
6FD2	625A		
6FD4	534B		
6FD6	463F		
6FD8	3B35		
6FDA	2F2C		
6FDC	2824		
6FDE	241F		

LOC	OBJ	LINE	SOURCE STATEMENT
6FE0	746A	514	DW 6A74H, 5860H, 4850H, 3D43H, 3282H, 282CH, 2125H, 1D21H
6FE2	6058		
6FE4	5048		
6FE6	433D		
6FE8	8232		
6FEA	2C28		
6FEC	2521		
6FEE	211D		
6FF0	7469	515	DW 6974H, 565FH, 454EH, 3940H, 2F35H, 2529H, 1E22H, 1B1EH
6FF2	5F56		
6FF4	4E45		
6FF6	4039		
6FF8	352F		
6FFA	2925		
6FFC	221E		
6FFE	1E1B		
		516	END

PUBLIC SYMBOLS

EXTERNAL SYMBOLS

USER SYMBOLS

AGAIN	A 64EC	ALPHA	A 6450	BAK	A 65BF	BINRY	A 6416	DAC1	A 64FE
DAC3	A 650A	DAC4	A 6510	DAC5	A 6516	DAC6	A 651C	DAC7	A 6522
DAK	A 64F6	FINIS	A 674A	GOON	A 66FB	INIT1	A 645D	INIT2	A 649A
IT	A 6493	M1	A 6000	M10	A 639B	M11	A 63BD	M2	A 6026
M4	A 6268	M5	A 6293	M6	A 62C7	M7	A 62F8	M8	A 630F
MSG	A 6678	MSG4	A 6628	NEXT	A 663C	NDALX	A 6551	NDALY	A 6567
OK1	A 6664	OK2	A 668C	P1P	A 6772	P3P	A 67F5	POSVE	A 65A7
STORE	A 65BA	TRW	A 643F	WAIT	A 6723	ZER	A 67BE	ZERO	A 657B

ASSEMBLY COMPLETE, NO ERRORS

98553

# Thesis

621-38411

R158m Date Slip

A 98555

This book is to be returned on the  
date last stamped.

[illegible]

EE-1987-M-RAN-MIC